VAX/VMS
Magnetic Tape User's Guide

Order No. AA-M539A-TE

May 1982

This manual describes magnetic tape formats and procedures for using magnetic tape on VAX/VMS Version 3.0.

REVISION/UPDATE INFORMATION: This is a new manual.

SOFTWARE VERSION: VAX/VMS Version 3.0

digital equipment corporation · maynard, massachusetts

First Printing, May 1982

Printed in U.S.A.

The postpaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DEC	DIBOL	RSX
DEC/CMS	EduSystem	UNIBUS
DECnet	IAS	VAX
DECsystem-10	MASSBUS	VMS
DECSYSTEM-20	PDP	VT
DECUS	PDT	digital
DECwriter	RSTS	

ZK2111

HOW TO ORDER ADDITIONAL DOCUMENTATION

In Continental USA and Puerto Rico call 800-258-1710

In New Hampshire, Alaska, and Hawaii call 603-884-6660

In Canada call 613-234-7726 (Ottawa-Hull)
800-267-6146 (all other Canadian)

DIRECT MAIL ORDERS (USA & PUERTO RICO)*

Digital Equipment Corporation
P.O. Box CS2008
Nashua, New Hampshire 03061

*Any prepaid order from Puerto Rico must be placed with the local Digital subsidiary (809-754-7575)

DIRECT MAIL ORDERS (CANADA)

Digital Equipment of Canada Ltd.
940 Belfast Road
Ottawa, Ontario K1G 4C2
Attn: A&SG Business Manager

DIRECT MAIL ORDERS (INTERNATIONAL)

Digital Equipment Corporation
A&SG Business Manager
c/o Digital's local subsidiary or approved distributor

Internal orders should be placed through the Software Distribution Center (SDC), Digital Equipment Corporation, Northboro, Massachusetts 01532

CONTENTS

CONTENTS

EXAMPLES

FIGURES

TABLES

CONTENTS

PREFACE

MANUAL OBJECTIVES

This manual describes magnetic tape formats and procedures for using magnetic tape on VAX/VMS Version 3.0.

INTENDED AUDIENCE

This manual is intended for all magnetic tape users who need to access, read, and write magnetic tape volumes on a VAX/VMS system. Although knowledge of the VAX/VMS operating system, DIGITAL Command Language (DCL), and magnetic tape concepts is helpful, this manual describes pertinent concepts and procedures for both the novice and the experienced magnetic tape user.

STRUCTURE OF THIS DOCUMENT

This manual consists of five chapters, four appendixes, and a glossary.

- Chapter 1 introduces basic magnetic tape concepts.
- Chapter 2 describes ANSI labels and their format supported by VAX/VMS, file and volume configurations, and record formats.
- Chapter 3 shows how to access tape files by using VAX-11 Record Management Services (VAX-11 RMS) or Queue I/O ($QIO) calls. This chapter also discusses accessing ANSI label facilities from high-level languages.
- Chapter 4 shows how to use ANSI labeled tapes on VAX/VMS.
- Chapter 5 shows how to use foreign tapes on VAX/VMS.
- Appendix A contains tables describing the fields in VAX/VMS ANSI volume, header, and trailer labels.
- Appendix B contains tables specifying initialization values of fields in VAX/VMS ANSI volume, header, and trailer labels.
- Appendix C contains guidelines for handling and selecting magnetic tape volumes.
- Appendix D lists the portions of the ANSI Level 3 standard that VAX/VMS does not support.
- The glossary defines terminology frequently used in this manual.

ASSOCIATED DOCUMENTS

Major documents relevant to topics discussed in this manual are listed below. For descriptions and order numbers of the VAX/VMS documents, refer to the VAX-11 Information Directory and Index.

- VAX-11 Record Management Services Reference Manual
- VAX/VMS I/O User's Guide
- VAX/VMS System Management and Operations Guide
- VAX/VMS System Services Reference Manual
- American National Standard for Magnetic Tape Labels and File Structure for Information Interchange (ANSI, X3.27-1978)
- American National Standard for Recorded Magnetic Tape for Information Interchange (800 CPI, NRZI), ANSI X3.22-1973
- American National Standard for Recorded Magnetic Tape for Information Interchange (1600 CPI, PE), ANSI X3.39-1973
- American National Standard for Recorded Magnetic Tape for Information Interchange (6250 CPI, Group-Coded Recording), ANSI X3.54-1976
- American National Standard for Unrecorded Magnetic Tape for Information Interchange (9-track 200 and 800 CPI, NRZI, and 1600 CPI, PE), ANSI X3.40-1976

CONVENTIONS USED IN THIS DOCUMENT

Convention	Meaning
quotation marks apostrophes	The term "quotation marks" refers to double quotation marks ("). The term "apostrophe" refers to a single quotation mark (').
[ ]	Square brackets indicate that the enclosed item is optional.
...	A horizontal ellipsis indicates that the preceding item(s) can be repeated one or more times. For example: file-spec[,file-spec...]
. . .	A vertical ellipsis indicates that not all of the statements in an example or figure are shown.

Convention	Meaning
CTRL/x	The symbol CTRL/x indicates that you must press the key labeled CTRL while you simultaneously press another key denoted by the variable x. For example, CTRL/C means that you press the CTRL key while pressing the C key. Because the system echos this control sequence as ^C, all examples illustrating responses from the system use the format ^x, where x is the variable key in a control sequence.

Unless otherwise noted, all numeric values are represented in decimal notation.

Unless otherwise specified, you invoke commands by pressing the RETURN key.

CHAPTER 1

INTRODUCTION

Magnetic tape is less expensive and easier to store than disks. For these reasons, magnetic tape has historically been used for archival storage, where backup copies of data bases have been kept against some natural disaster such as fire.

Magnetic tapes are also easy to carry, so they have been used to transfer data from one computer system to another.

Because magnetic tape is a sequential medium, you cannot perform random access to the data that resides on a magnetic tape. Because magnetic tape is nonshareable, only one process can access the data at a time. Furthermore, access to data on magnetic tape is often much slower than access to data on disk. Of course, many applications only need single sequential access, and in some cases magnetic tapes can be used instead of disks.

1.1 MAGNETIC TAPE VOLUMES AND FILES

A reel of magnetic tape is referred to as a magnetic tape **volume**. A magnetic tape **volume set** consists of one or more volumes.

A magnetic tape file is simply a related collection of bytes of data. A magnetic tape file can reside on one volume, or the file can continue on additional volumes. The data can be set off and labeled by predetermined fields, or the data can be unlabeled. VAX/VMS supports Level 1, Level 2, and much of Level 3 of the ANSI standard for magnetic tape labels and file structure. Because the ANSI standards are widely accepted, using ANSI-labeled magnetic tape volumes makes it easier to transfer data between different computer systems.

If the data is unlabeled, or if it is labeled according to some conventions other than the ANSI standards, the magnetic tape volume is referred to as a **foreign** tape volume. VAX/VMS can access the data on a labeled foreign tape by simply treating the labels as bytes of data.

1.2 DENSITY

Different tape drives write magnetic tapes at different densities. The ANSI standards define density as the number of characters per inch (cpi) recorded on a volume. VAX/VMS uses the equivalent term bytes per inch (bpi).

The encoding method that a magnetic tape drive uses to record data on a volume determines the volume density. VAX/VMS drives and the ANSI standards support the following recording methods: 6250 BPI

- Non-return-to-zero-inverted (NRZI), which records data at 800 bpi
- Phase-encoded (PE), which records data at 1600 bpi
- Group-coded recording (GCR), which records data at 6250 bpi

These recording methods are described in the ANSI standards. The magnetic tape drives, the methods, and the densities supported by VAX/VMS are listed in Table 1-1.

Table 1-1: Magnetic Tape Drives, Recording Methods, and Densities Supported by VAX/VMS

Type of Magnetic Tape Drive	Recording Method	Density (bpi)
TE16	NRZI	800
	PE	1600
TS11	PE	1600
TU45	NRZI	800
	PE	1600
TU77	NRZI	800
	PE	1600
TU78	PE	1600
	GCR	6250

For you to access, read, or write data on a volume, the magnetic tape drive on which the volume is mounted must record data in the same density at which the volume was recorded previously. For example, if you try to mount a volume recorded at 800 bpi on a magnetic tape drive that operates only at 1600 or 6250 bpi, your process will receive an error status message. (For details on VAX/VMS error status messages, refer to the VAX/VMS System Messages and Recovery Procedures Manual.)

VAX/VMS tape drives usually default to the density at which the first record on the volume was written if the drive supports that density. However, there are exceptions. VAX/VMS tape drives will always default to a density of 1600 bpi if you are initializing a new, unformatted volume that either has never been written or that has been processed by a verifying machine. The drives will also default to 1600 bpi if you are using the /FOREIGN or the /NOLABEL qualifier with the MOUNT command and the first operation performed on the volume is a write operation.

1.3 PERFORMING INPUT/OUTPUT ON MAGNETIC TAPES

You can access files on VAX/VMS ANSI-labeled volumes by using virtual, logical, or physical input/output (I/O) requests. You cannot use virtual I/O with foreign volumes.

1.3.1 Virtual I/O

Virtual I/O allows you to use files to access, read, or write data on volumes. You request virtual I/O by default when you open a file using VAX-11 Record Management Services (VAX-11 RMS) or DCL commands. You also can specify virtual I/O explicitly by using Queue I/O ($QIO) system service calls.

Because virtual I/O requires a file structure, you cannot use it with foreign volumes. (If you attempt a virtual I/O operation with a foreign volume, the system will default to logical I/O.) This means that you cannot perform file-oriented operations with a foreign magnetic tape volume. You can, however, perform record-oriented operations.

1.3.2 Logical I/O

Logical I/O allows you to access, read, or write blocks of data. To specify logical I/O you must either explictly open a file with VAX-11 RMS and perform block I/O or use $QIO calls to specify block I/O directly. Most DCL commands and VAX-11 RMS functions will default to logical I/O for foreign volumes; each block of data is treated as a record.

If you know the block structure of a foreign volume, you can use logical I/O with it. If the block structure is unknown, you must use physical I/O.

1.3.3 Physical I/O

Physical I/O allows you to access, read, or write data at relative block locations on a volume. This type of access allows you to bypass any format on a magnetic tape volume. To specify physical I/O, you must use $QIO calls.

1.4 ASCII "A" CHARACTER SUPPORT

VAX/VMS ANSI-labeled magnetic tape volumes support the full range of ASCII "a" characters for logical names and VAX/VMS ANSI labels, which include volume identifiers (volume names) and file identifiers (file names) described in Chapter 2. A subset of the ASCII character set, the set of "a" characters consists of the characters listed below.

- Digits 0 through 9
- Uppercase letters A through Z
- Special characters:
  = ! % " & ( ) * + - , . : ; < > ' ? space

To specify ASCII "a" characters in logical names, volume identifiers, or file identifiers, you must enclose the character string in a pair of quotation marks (""). Lowercase characters are not legal ASCII "a" characters, but if you specify them, VAX/VMS changes them to uppercase.

CHAPTER 2

VAX/VMS ANSI-LABELED MAGNETIC TAPE

This chapter describes ANSI labels, data, and record formats supported by VAX/VMS.

2.1 LOGICAL FORMAT OF ANSI-LABELED VOLUMES

The format of VAX/VMS ANSI-labeled magnetic tape volumes is based on Level 3 of the ANSI standard for magnetic tape labels and file structure for information interchange. This standard specifies the format, content, and sequence of volume labels, file labels, and file structures. According to this standard, volumes are written and read on 9-track magnetic tape drives only. The contents of labels must conform to prescribed data and record formats. All labels must consist of ASCII "a" characters.

The VAX/VMS ANSI-labeled format allows you to write binary data in the file sections (see Figure 2-1) of files. However, if you plan to use such files for information interchange across systems, make sure that the recipient system can read the binary data.

2.2 VAX/VMS MAGNETIC TAPE ANCILLARY CONTROL PROCESS (MTAACP)

The VAX/VMS magnetic tape ancillary control process (MTAACP) is the internal software process of the operating system that interprets the logical format of VAX/VMS ANSI-labeled volumes. Transparent to your process, the MTAACP reads, writes, and interprets VAX/VMS ANSI labels before passing this information to VAX-11 Record Management Services (VAX-11 RMS) and Queue I/O ($QIO) system services. These services in turn read, write, and interpret the record format of the data written in the file section.

2.3 BASIC COMPONENTS OF THE VAX/VMS ANSI-LABELED FORMAT

The format of VAX/VMS ANSI-labeled magnetic tape consists of four basic components: beginning-of-tape and end-of-tape markers, tape marks, file sections, and volume, header, and trailer labels. Figure 2-1 displays the arrangement and function of these components.

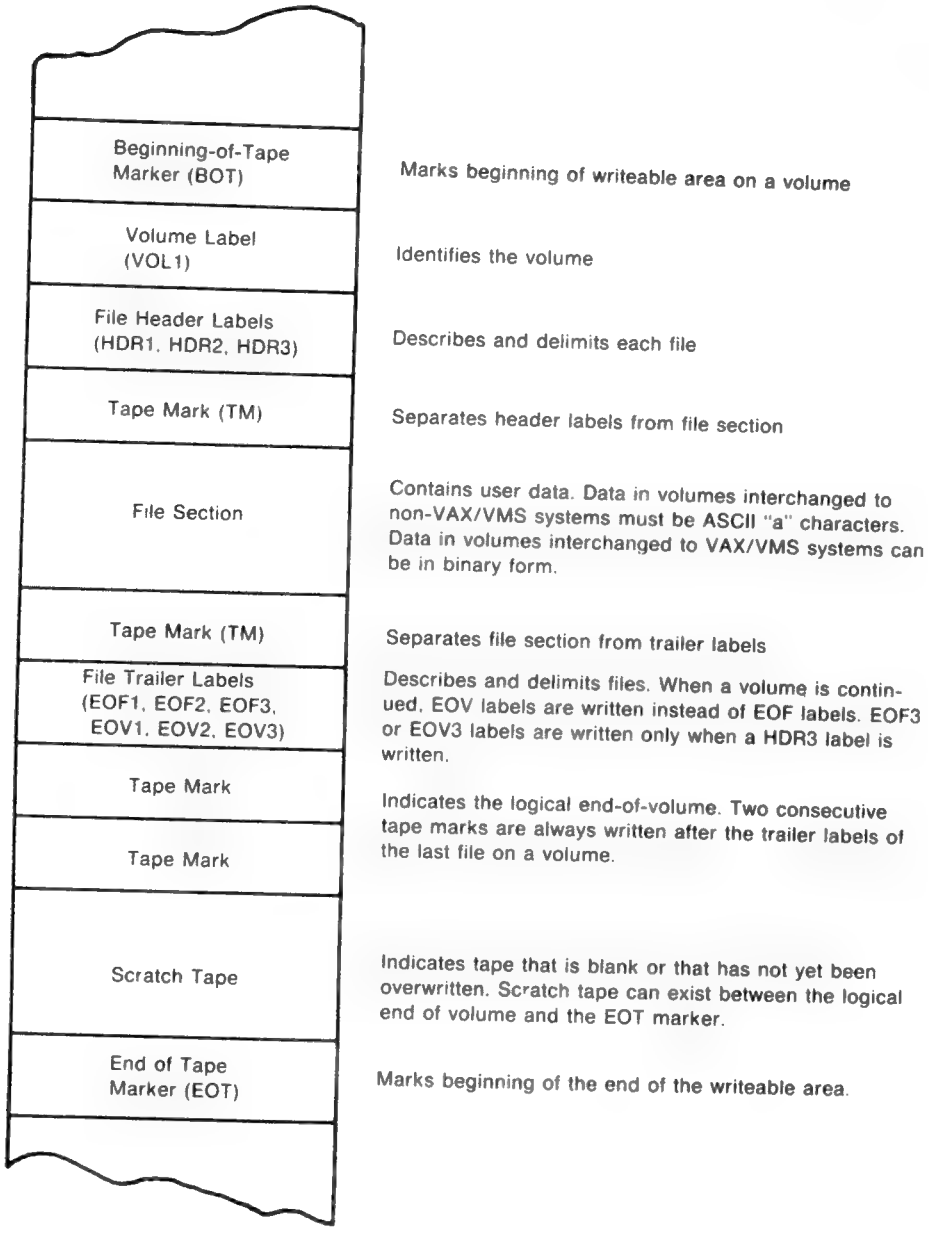


Figure 2-1: Basic Layout of a VAX/VMS ANSI-Labeled Volume

2.3.1 Beginning-Of-Tape and End-Of-Tape Markers

Every volume has beginning-of-tape (BOT) and end-of-tape (EOT) markers. These markers are pieces of photoreflective tape that delimit the writeable area on a volume. ANSI standards require that a minimum of 14 feet to a maximum of 18 feet of magnetic tape precede the BOT marker; a minimum of 25 feet to a maximum of 30 feet of magnetic tape, of which 10 feet must be writeable, must follow the EOT marker. The EOT marker indicates the beginning of the end of the writeable area of the tape, rather than the physical end of the tape. Therefore, data and labels can be written after the EOT marker.

2.3.2 Tape Marks

Tape marks separate the file labels from the file sections, separate one file from another, and denote the logical end-of-volume. On the basis of the number and relative placement of tape marks written on a volume, VAX/VMS determines whether a tape mark delimits a label, a file, or a volume.

Tape marks are written both singly and in pairs. Single tape marks separate either a file section from the header and trailer labels or one file from another. When written after a set of header labels, a single tape mark signals the beginning of a file section. When written before a set of trailer labels, a single tape mark indicates the end of a file section. When written after a trailer label set, a single tape mark separates one file from another.

Double tape marks indicate that either an empty file section exists or that the logical end-of-volume has been reached. A VAX/VMS system creates an empty file when a volume is initialized. For example, when double tape marks are placed between header and trailer labels, the tape marks indicate that an empty file section exists. When double tape marks are written after the trailer labels of the last file on a volume, they indicate the logical end-of-volume.

2.3.3 Labels

Labels identify, describe, and control access to volumes and their files. VAX/VMS ANSI-labeled format supports volume, header, and trailer labels. The volume label is the first label written on a volume. It identifies the volume and the volume owner and specifies access protection. Header and trailer labels are sets of labels that identify, describe, and delimit files. Header labels precede files; trailer labels follow files.

Table 2-1 lists the labels supported by VAX/VMS. All other ANSI labels are ignored by VAX/VMS on input.

Although each type of label uses a different format to organize its contents, all labels must consist of ASCII "a" characters. Some labels contain reserved fields designed for future system use or future ANSI standardization. Reserved fields also must consist of ASCII "a" characters; however, VAX/VMS ignores these characters on input.

2.4 VOLUME AND FILE CONFIGURATIONS

VAX/VMS ANSI-labeled volumes support four file/volume configurations: single file/single volume; single file/multivolume; multifile/single volume; and multifile/multivolume. All these configurations conform to the following guidelines:

- The file sequence number field allows as many as 9999 file sections for one file. In effect, the file length is unlimited.
- Only one file section of a given file is written on a volume.
- When multiple sections exist for one file, each file section is written to a separate volume in the volume set. The file section numbers of each section are written consecutively in

ascending order (section n+1 is written immediately following section n); file sections of other files are not interspersed.

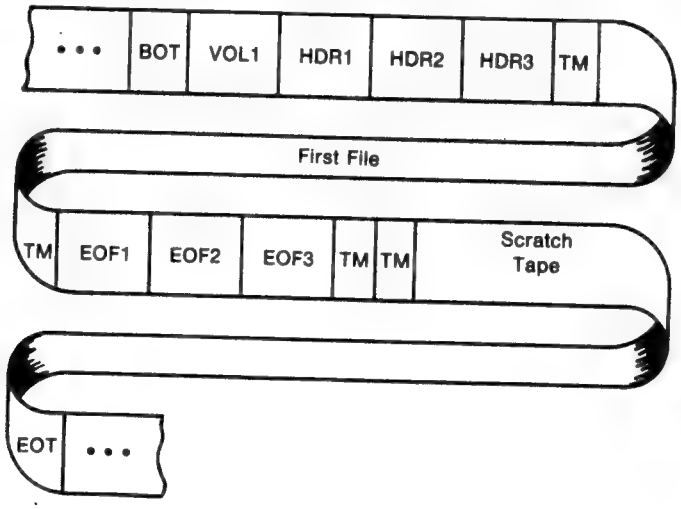
Each of the file/volume configurations is illustrated in the subsections that follow.

Table 2-1: Labels and Components Supported by VAX/VMS

Symbol	Meaning	Symbol	Meaning
BOT	Beginning-of-tape marker	EOV3	Third end-of-volume label
EOF1	First end-of-file label	HDR1	First header label
EOF2	Second end-of-file label	HDR2	Second header label
EOF3	Third end-of-file label	HDR3	Third header label
EOT	End-of-tape marker	VOL1	Volume label
EOV1	First end-of-volume label	TM	Tape mark
EOV2	Second end-of-volume label	TM TM	Double tape mark indicates an empty file section or the logical end-of-volume

2.4.1 Single File/Single Volume Configuration

A single file/single volume configuration consists of one file on one volume. The components of the ANSI-labeled format for this configuration are illustrated in Figure 2-2.

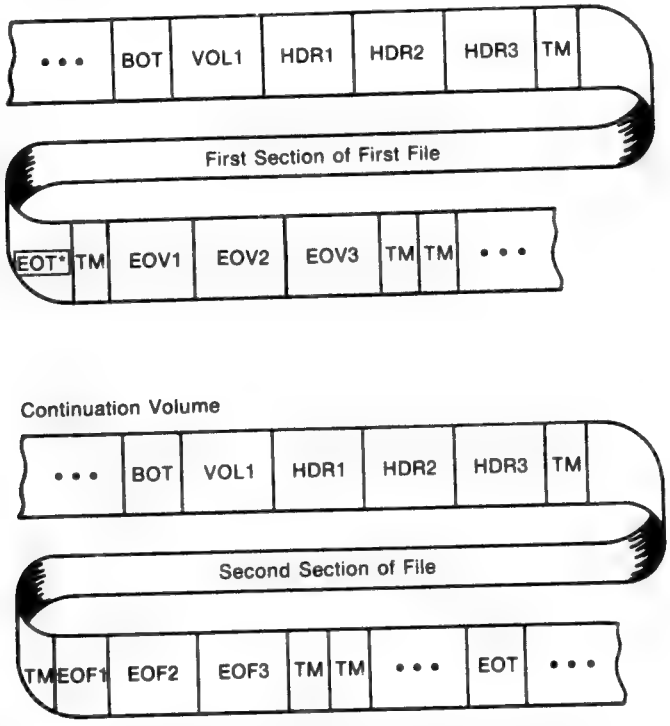


ZK-346-81

Figure 2-2: Single File/Single Volume Configuration

2.4.2 Single File/Multivolume Configuration

A single file/multivolume configuration consists of one file that spans two or more volumes in a volume set. Figure 2-3 illustrates the components of the ANSI-labeled format for this configuration.



*When the driver encounters an EOT marker during a write operation, the MTAACP writes the appropriate trailer labels and performs a volume switch, if necessary.

ZK-347-81

Figure 2-3: Single File/Multivolume Configuration

2.4.3 Multifile/Single Volume Configuration

A multifile/single volume configuration consists of two or more files on a single volume. It is the most common file and volume configuration. Figure 2-4 illustrates the components of the ANSI-labeled format for this configuration.

2.4.4 Multifile/Multivolume Configuration

A multifile/multivolume configuration consists of two or more files that span two or more volumes in the same volume set. Figure 2-5 illustrates the components of the ANSI-labeled format for this configuration.

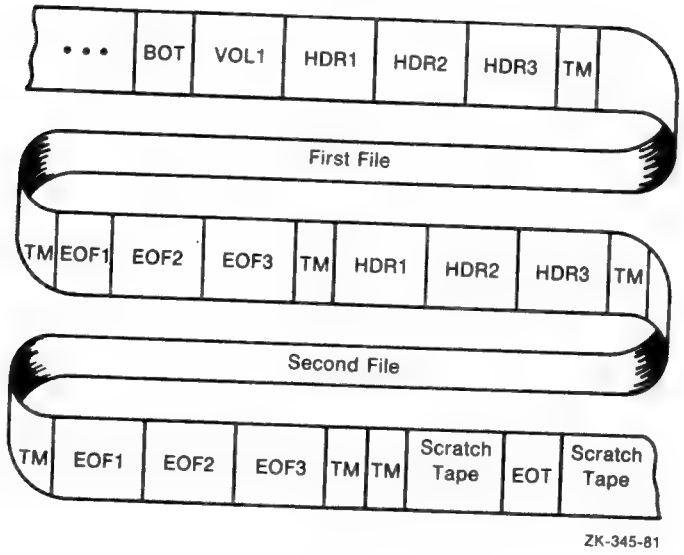


Figure 2-4: Multifile/Single Volume Configuration

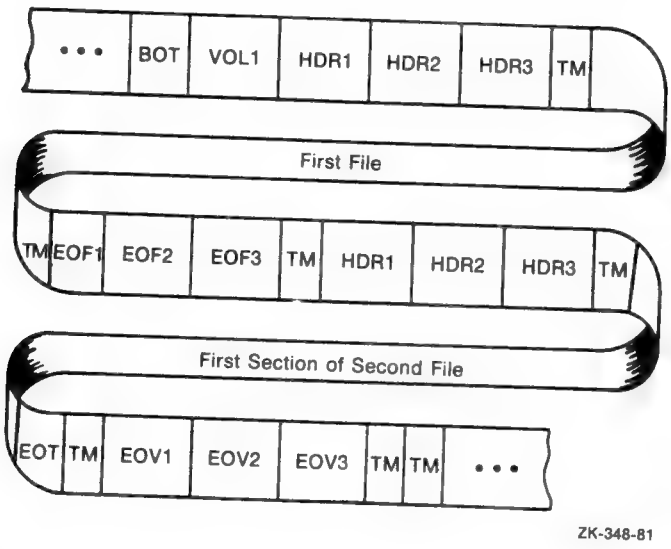


Figure 2-5: Multifile/Multivolume Configuration

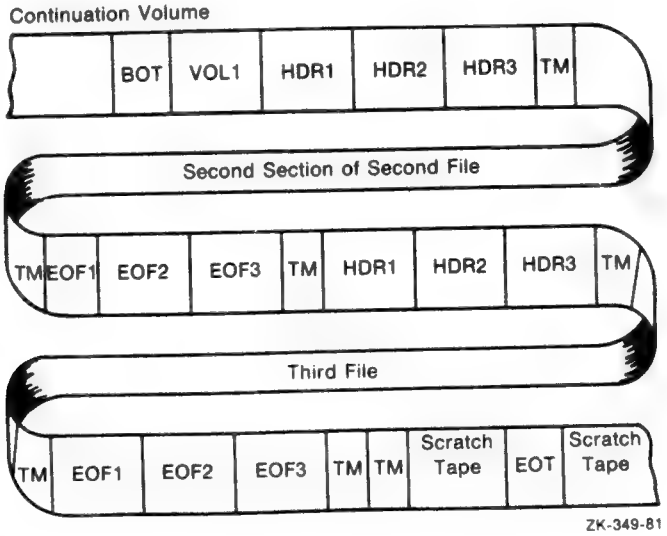


Figure 2-5 (Cont.): Multifile/Multivolume Configuration

2.5 VOLUME LABEL

The 80-character volume label (VOL1) is the first label written on a VAX/VMS ANSI-labeled volume. It defines the label type, name, and owner of the volume (see Figure 2-6). Although there are many fields in a VOL1 label, this section describes only those fields that you can access or that can inhibit access to a volume and its files. All the fields in the VOL1 label are described in Appendix A.

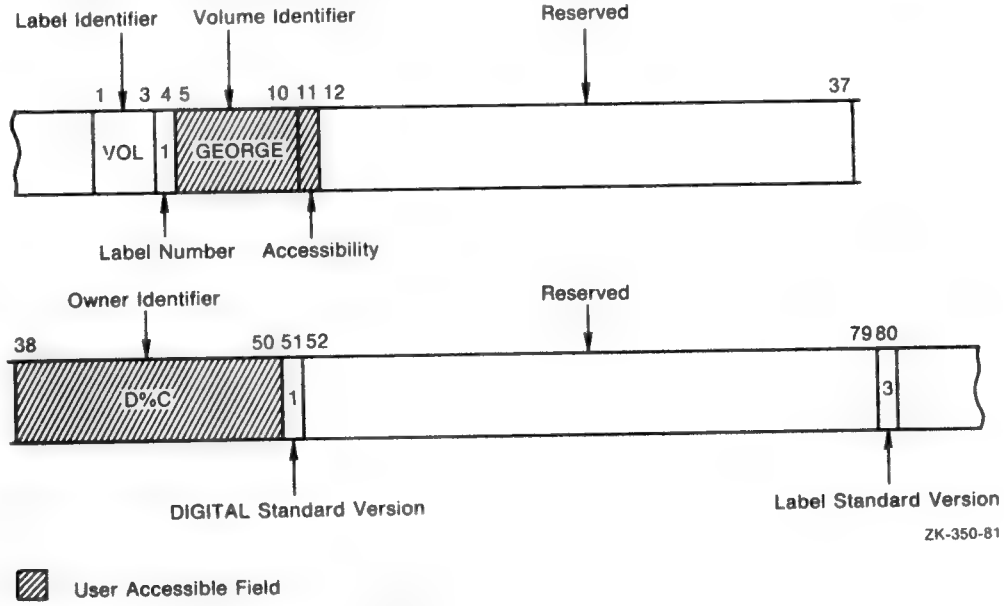


Figure 2-6: VAX/VMS ANSI VOL1 Label Format

2.5.1 Volume Identifier Field

The volume identifier field is a 6-character field that contains the name of the volume. You specify the volume identifier in the command

string when you initialize or mount a volume (see Chapter 4). In that context, the volume identifier is a volume label. In fact, it is only part of the VAX/VMS ANSI VOL1 label. In this manual, the term "volume identifier" refers to the field within the VOL1 label.

The volume identifier consists of six ASCII "a" characters. Lowercase characters are not in the "a" set, but if you specify them, VAX/VMS will change them to uppercase. If you specify fewer than six characters, VAX/VMS pads the field by writing spaces to the right of any specified characters.

2.5.2 Accessibility Field

The accessibility field is a 1-character field that controls access to a volume. Neither access protection code nor a user identification code (UIC) denoting volume ownership is written in this field. When this field contains any character other than a space, VAX/VMS does not allow your process to access the volume and returns this message: magnetic tape file access is nonblank. Your process can access the volume if you specify the /OVERRIDE=ACCESSIBILITY qualifier when you mount or initialize it.

By default, VAX/VMS writes a space to this field. You can, however, write any ASCII "a" character in this field if you specify the /LABEL=VOLUME_ACCESSIBILITY qualifier with the INITIALIZE command.

You must have VOLPRO privilege to specify the /OVERRIDE=ACCESSIBILITY qualifier, if the volume is a VAX/VMS ANSI-labeled volume, if you are not a SYSTEM user or the volume owner, and protection on the volume restricts your process from accessing the volume.

2.5.3 Owner Identifier Field

The owner identifier field identifies the DIGITAL system on which the volume was created and specifies the access protection code for the volume.

The first three characters (38, 39, and 40) inform the VAX/VMS MTAACP and the Mount program which DIGITAL system formatted the volume, thus enabling the MTAACP to interpret the data in the field correctly. These characters are as follows:

Character	Contents
38-39	D%
40	System identifier code(x)

The system identifier code for VAX/VMS is an uppercase C.

The owner identifier field also contains a value that incorporates the user identification code (UIC) with the access protection code specified for a volume. By default, VAX/VMS does not write a UIC to this field, thus allowing all users read and write access. You can specify read and write access only; execute and delete access are not applicable to magnetic tape volumes. In addition, regardless of the access protection code(s) that you specify, both system users and the volume owner always have read and write access to a volume. The owner identifier field can contain any one of the values shown in Table 2-2. The contents of this field depends on the access protection code that you specify.

Table 2-2: Protection Codes in the VAX/VMS Owner Identifier Field

Group	Privilege	Character Positions	Contents	Code Specified[1]
SYSTEM OWNER	Read/write Read/write	41 - 45 46 - 50	5-character ASCII group number 5-character ASCII member number	(G,W)
SYSTEM OWNER GROUP	Read/write Read/write Read	41 - 45 46 47 - 50	5-character ASCII group number Character A 4 characters of member number	(G:R,W)
SYSTEM OWNER GROUP WORLD	Read/write Read/write Read Read	41 42 - 45 46 47 - 50	Character A 4 characters of group number Character A 4 characters of member number	(G:R,W:R)
SYSTEM OWNER GROUP	Read/write Read/write Read/write	41 - 45 46 - 50	5-character ASCII group number Spaces	(G:RW,W)
SYSTEM OWNER GROUP WORLD	Read/write Read/write Read/write Read	41 42 - 45 46 - 50	Character A 4 characters of group number Spaces	(G:RW,W:R)
SYSTEM OWNER GROUP WORLD	Full privileges	41 - 50	Spaces (default)	(G:RW,W:RW)

1. Code specified replacing the ellipsis in the parentheses of the DCL command: INITIALIZE/PROTECTION = (...).

2.6 HEADER LABELS

VAX/VMS supports three file header labels: HDR1, HDR2, and HDR3. The HDR3 label is optional. The following sections describe and illustrate each file header label.

2.6.1 HDR1

Every file on a volume has a HDR1 label, which identifies and describes the file by supplying the VAX/VMS MTAACP with the following information:

- File identifier
- File-set identifier
- File section number
- File sequence number
- Generation and generation version numbers
- File creation and expiration dates
- Accessibility code
- System code

The components of the HDR1 label are displayed in Figure 2-7. Although the HDR1 label contains many fields, only those fields that you can access, override, or concern file access are discussed in this section. Appendix A describes all the fields in the HDR1 label.

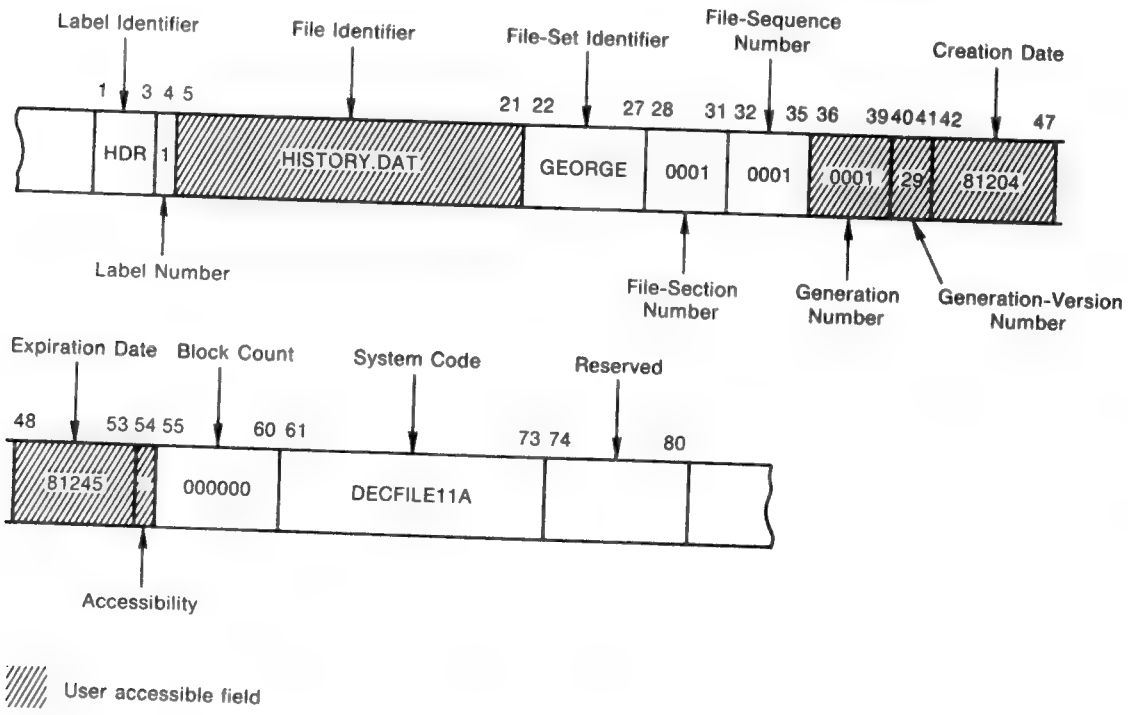


Figure 2-7: HDR1 Label Format

2.6.1.1 File Identifier Field - The file identifier field contains the file name that you specify. You may use either an ANSI file name or a partial VAX/VMS file specification of the following format:

```
filename.type;version
```

VAX/VMS file specifications are a subset of ANSI file names. However, ANSI file names are valid only for magnetic tape volumes; VAX/VMS file specifications are valid for disk and tape volumes. Both types of file specifiers are compatible with compatibility mode.

A partial VAX/VMS file specification consists of a file name, a file type, and an optional version number. Valid file names contain a maximum of nine characters. Valid file types consist of a period followed by a maximum of three characters. The semicolon separates the version number from the file type.

Except for wild card characters, only the characters A through Z and 0 through 9 are valid for VAX/VMS file names and types. The period and semicolon are the only valid special characters, and they are always separators.

ANSI file names do not have a file type field. An ANSI file name consists of a 17-character name string, a period, a semicolon, and an optional version number. You can specify a name string consisting of a maximum of 17 ASCII "a" characters, but you must enclose the string in a pair of quotation marks (as in, "file name"). When you specify fewer than 17 characters, the string is padded on the right with spaces to the 17-character maximum size. If you specify a file name that has trailing spaces, VAX/VMS truncates them when the file name is returned. In addition, the space-padded field prevents you from specifying a unique file name that consists of spaces.

The quotation mark character requires special treatment because it is both the file name delimiter and a valid ASCII "a" character that can itself be embedded in the name string. You must specify two quotation marks for each one that you want VAX/VMS to interpret. The additional quotation mark informs VAX/VMS that one of the quotation marks is part of the name string, rather than a delimiter.

Embedded spaces also are valid characters, but embedded tabs are not. Lowercase characters are not in the ASCII "a" character set but if you specify them, VAX/VMS converts them to uppercase characters.

If you do not specify a file type or version number on input, VAX/VMS will supply a period (the default file type) and a semicolon (the default version number). However, the period and semicolon will not be written to this field on the tape.

Although VAX/VMS considers version numbers for ANSI and VAX/VMS file names to be part of the file name specification, the version number of a file is not written to the file identifier field, but is mapped to the generation number and generation version number fields as described in Section 2.6.1.3.

Examples below display ANSI file names. The input is the format that you specify. The output shown displays the VAX/VMS format returned to your process and the format written to the label. The number sign (#) in the examples below indicate a space character. In the last example, a VAX/VMS file name is enclosed in quotation marks, like an ANSI file name, on input. However, VAX/VMS returns the file name to the process as a VAX/VMS filename, rather than an ANSI filename. Therefore, when you enclose a valid VAX/VMS file name in quotation marks on input, VAX/VMS parses the file name as a VAX/VMS file name.

Input	Output to User Process	Output to HDR1 Label
"AB2&D""FgHI*k4""#-M".;2	"AB2&D""FGHI*K4""#-M".;2	AB2&D"FGHI*K4"#-M
"##########"	"".;	##################
**********;	**********.;	###############
"DWDEVOP.DAT"	DWDEVOP.DAT;	DWDEVOP.DAT######

2.6.1.2 File-Set Identifier Field - The 6-character file-set identifier field denotes all files that belong to the same volume set. The file-set identifier for any file within a given volume set should always be the same as the file-set identifier of the first file on the first volume that you mount. For VAX/VMS, the file-set identifier is the same as the volume identifier of the first volume that you mount. Thus, when you specify the volume identifier for the first volume that you mount, you indirectly specify the file-set identifier for all files within the volume set.

2.6.1.3 Generation Number and Generation-Version Number Fields - The generation number and generation-version number fields store the file version number that is specified on input and written by VAX/VMS on output. VAX/VMS does not increment the version number of a file, even when the version of the specified file already exists on the volume. Therefore, if the file that you specify has the same file name and version number as an existing file, you will have at least two files with the same version number on the same volume set.

On input, VAX/VMS computes the version number by using this calculation:

version number =
[(generation number - 1)100] + generation-version number + 1

Version numbers larger than 32767 are divided by 32768; the integer remainder becomes the version number.

On output, the generation number is derived from the version number with this calculation:

generation number = [(version number - 1)/100] + 1

If there is a remainder after the version number is divided by 100, the remainder becomes the generation-version number. It is not added to 1 to form the generation number.

2.6.1.4 Creation Date and Expiration Date Fields - The creation date field contains the date the file is created. The expiration date field contains the date the file expires. VAX/VMS interprets the expiration date of the first file on a volume as the date that both the file and the volume expire. VAX/VMS stores the creation and expiration dates in the Julian format. This 6-character format (#YYDDD) consists of a space (#), the year, and the day. Only dates are relevant for these fields; time is always returned as 00:00:00:00.

By default, VAX/VMS writes the current date to both the creation and expiration date fields when you create a file. Because the expiration date is the same as the creation date, the file expires on creation and you can overwrite it immediately. If the expiration date is a date that is later than the creation date and if the files that you want to overwrite have not expired, you must specify the /OVERRIDE=EXPIRATION qualifier with the INITIALIZE or MOUNT command.

To write dates other than the VAX/VMS defaults in the date fields in this label, specify the creation date field (CDT) and the expiration date field (EDT) of the VAX-11 RMS date and time extended attribute block (XABDAT).

When you do not specify a creation date, VAX-11 RMS defaults the current date to the creation date field.

To specify a zero creation date, you must specify a year before 1900. If you specify a binary zero in the date field, the system will write the current date to the field.

For details on the XABDAT, refer to the VAX-11 Record Management Services Reference Manual.

2.6.1.5 Accessibility Field - The accessibility field in the HDR1 label can prevent your process from accessing a file. Its function is similar to the accessibility field in the VOL1 label. By default, VAX/VMS writes a space in this field. If this field contains any character other than a space, VAX/VMS returns this error status message when you access the file: magnetic tape file access is nonblank.

Specifying the /OVERRIDE=ACCESSIBILITY qualifier when you mount or reinitialize a volume allows you to override the volume and file protection. You must have VOLPRO privilege to use this qualifier when the following conditions exist:

- The volume is a VAX/VMS ANSI-labeled volume.
- You are not the volume owner or a system user.
- The protection written on the volume restricts your process from accessing the volume (see Section 2.4.3).

This qualifier and procedures for initializing and mounting ANSI-labeled volumes are described in Chapter 4.

You can set the accessibility field by specifying an ASCII "a" character in the magnetic tape accessibility (MTACC) field of the VAX-11 RMS protection extended attribute block (XABPRO). For more details, refer to the VAX-11 Record Management Services Reference Manual.

2.6.2 HDR2 LABEL

A HDR2 label describes the record format, maximum record size, and maximum block size of a file. The components of a HDR2 label are shown in Figure 2-8. Although the HDR2 label contains many fields, only those fields that you can access or that concern file access are described in this section. Appendix A describes all the fields in a HDR2 label.

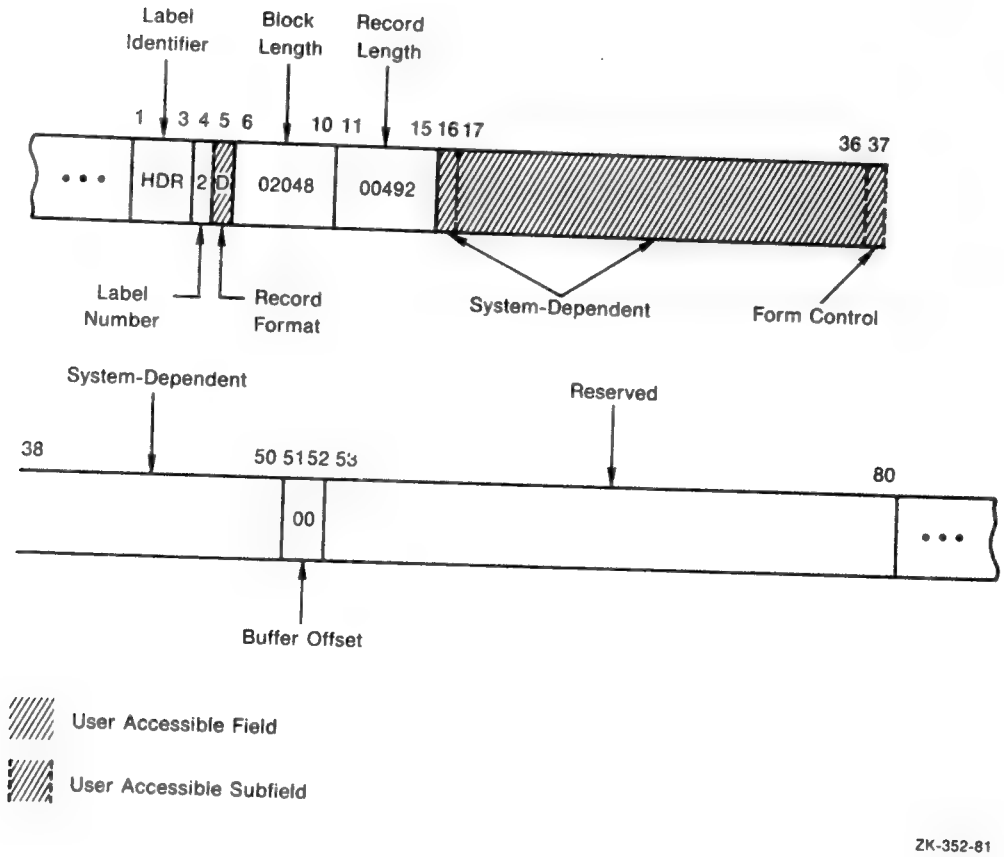


Figure 2-8: HDR2 Label Format

2.6.2.1 **Record Format Field** - The record format field specifies the type of record format the file contains. VAX/VMS supports two record formats: fixed length (F) and variable length (D). VAX/VMS does not support the spanned record format (S). When files contain record formats that VAX/VMS does not support, VAX/VMS cannot interpret the formats and classifies them as undefined.

Fixed-length records are all the same length. No indication of the record length is required within the records because either the HDR2 label defines the record length, or you specify the record length with the /RECORDSIZE qualifier.

A fixed-length record can be a complete block, or several records can be grouped together in a block.

When records are blocked, it is not necessary that the block be filled with data. If a block is not filled, it will be padded with caret characters (^). This means that you cannot have a fixed-length record containing only carets; the system will interpret this as padding, not data.

Figure 2-9 shows a block of fixed-length records. Each record has a fixed length of 50 bytes. All six records are contained in a 300-byte block. The records are blocked--that is, grouped together as one entity--to increase processing efficiency; when records are blocked, you can access many of them with one I/O request. The block size should be a multiple of the record size.

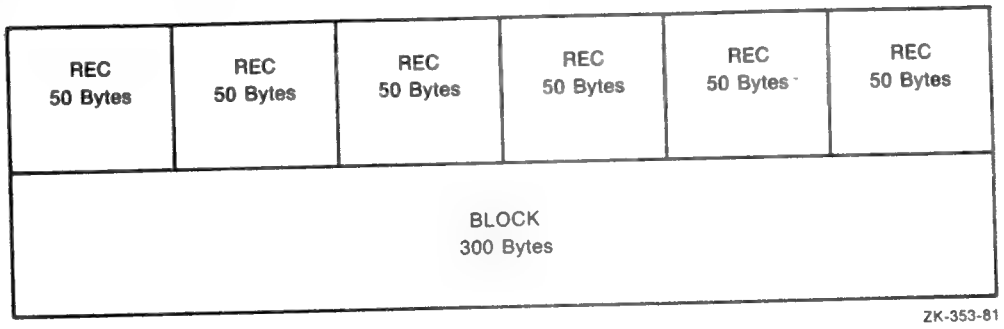


Figure 2-9: Blocked Fixed-Length Records

The size of a variable-length record is indicated by a record control word (RCW). The RCW consists of four bytes at the beginning of each record. A variable-length record can be a complete block, or several records can be grouped together in a block.

Two variable-length records are shown in Figure 2-10. The first consists of 54 bytes, including the RCW. The second consists of 112 bytes, including the RCW. The records are contained in a 166-byte block.

Do not use system-dependent record formats on volumes used for information interchange. VAX/VMS system-dependent formats are stream and variable with fixed control (VFC).

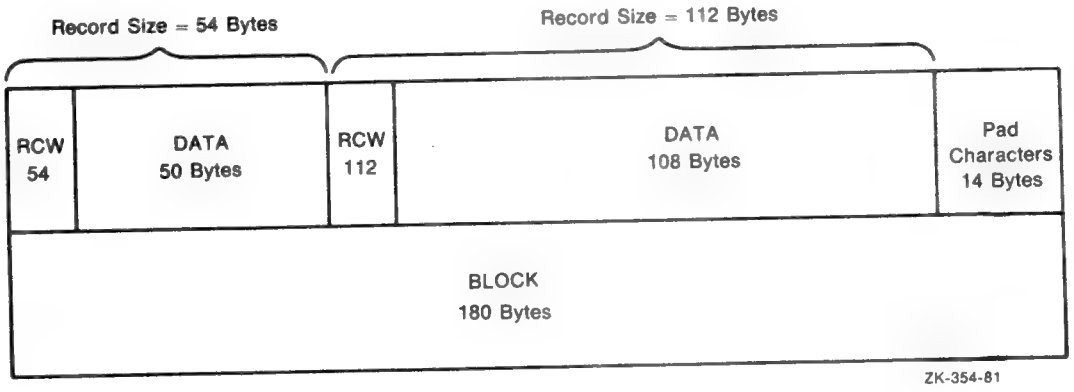


Figure 2-10: Variable-Length Records

2.6.2.2 Block Length Field - The block length field denotes the maximum size of the blocks. According to the ANSI standard, valid block sizes range from 18 to 2048 bytes. However, VAX/VMS allows you to specify a smaller or larger block size using the /BLOCKSIZE qualifier with the MOUNT command (see Section 4.3). To specify the block size using VAX-11 RMS, refer to the BLS field in the file access block (FAB) in the VAX-11 Record Management Services Reference Manual. When you specify a block size that is outside the ANSI standard range, the volume may not be processed correctly by other systems that support the ANSI standards.

2.6.2.3 Record Length Field - The record length field denotes either the size of fixed-length records or the maximum size of variable-length records in a file. Valid VAX/VMS record sizes vary, depending on the record format. The range for fixed-length records is 1 to 65534 bytes; the range for variable-length records is 4 to 9999 bytes, including the 4-byte RCW. Therefore, the maximum length of the data area of a variable-length record is 9995 bytes. To comply with ANSI standards, the record size should not be larger than the maximum block size of 2048 bytes, even though VAX/VMS allows larger record sizes (when the block size is larger).

For volumes containing files that do not have HDR2 labels, you must specify MOUNT/RECORDSIZE=n at mount time. The variable n denotes the record length in bytes. Files without HDR2 labels were created by a system that supports only Level 1 or 2 of the ANSI standard for magnetic tape labels and file structure. Records should be fixed length because this is the only record format that either level supports. If you do not specify a record size, each block will be considered a single record.

2.6.2.4 System-Dependent Field - The system-dependent field contains two 1-character subfields that describe how VAX/VMS interprets record format and form control.

The first subfield, character position 16, denotes whether the VAX-11 RMS attributes are in this label or the HDR3 label. If character position 16 contains a space, the VAX-11 RMS attributes are in the HDR3 label; if it contains any character other than a space, character position 16 is the first byte of the VAX-11 RMS attributes in the HDR2 label. The attributes appear in character positions 16 through 36 and 38 through 50.

For volumes created by any version of VAX/VMS through Version 2.0, the system-dependent fields of the HDR2 label contain the VAX-11 RMS file attributes in binary format. For volumes created by VAX/VMS Version 2.1 and subsequent versions, the HDR3 label stores the VAX-11 RMS attributes. Appendix A describes the fields in these labels.

The second subfield, the form control field at character position 37, specifies the form control that defines the carriage control applied to records within a file. Possible values supported for VAX/VMS magnetic tape volumes are listed below.

A	First byte of record contains FORTRAN control characters.
M	The record contains all form control information.
space	Line-feed/carriage-return combination is to be inserted between records when the records are written to a carriage control device, such as a line printer or terminal. If form control is not specified when a file is created, this is the default.

2.6.2.5 Buffer-Offset Length Field - For systems that support buffer offsets, the buffer-offset length field indicates the length of information that prefixes each data block. VAX/VMS, however, does not support buffer offsets and writes zeros to this field. If you mount and read a volume containing files that use nonzero buffer offsets on a VAX/VMS system, VAX/VMS interprets the information prefixing each data block as data. Thus, the data retrieved from volumes that implement buffer offsets is not predictable.

2.6.3 HDR3 LABEL

A HDR3 label describes the VAX-11 RMS file attributes. For VAX-11 RMS operations, data in the HDR3 label supersedes data in the HDR2 label.

Although a HDR3 label usually exists for every file on a VAX/VMS ANSI-labeled volume, there are two situations in which this label will not be written. The first is when an empty dummy file is created during volume initialization; no HDR3 label is written because the empty file does not require VAX-11 RMS attributes. The second is when you specify MOUNT/NOHDR3 at mount time. You should use the /NOHDR3 qualifier when you create files on volumes that will be interchanged to systems that do not process HDR3 labels correctly.

Figure 2-11 shows a HDR3 label. Although the HDR3 label contains many fields, only the VAX-11 RMS attributes field is described in this section.

2.6.3.1 The VAX-11 RMS File Attributes Field - The VAX-11 RMS attributes describe the record format of a file. These attributes are converted from 32 bytes of binary values to 64 bytes of ASCII representations of their hexadecimal equivalents for storage in the HDR3 label.

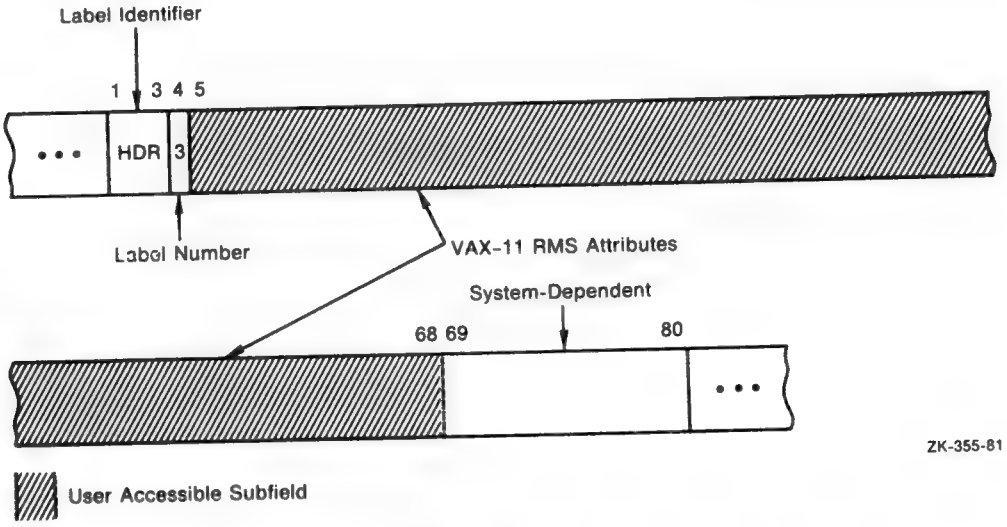


Figure 2-11: VAX/VMS ANSI HDR3 Label Format

2.7 TRAILER LABELS

VAX/VMS supports two sets of trailer labels: end-of-file (EOF) and end-of-volume (EOV). A trailer label is written for each header label.

EOF and EOV labels are identical to their file header label counterparts except that:

- The label identifier field (characters 1-3) contains EOF or EOV.
- The block count field (characters 55-60) in the EOF1 and EOV1 labels contains the number of data blocks in the file section.

The particular label that VAX/VMS writes depends on whether a file extends beyond a volume. If a file terminates within the limits of a volume, EOF labels are written to delimit the file (see Figure 2-2). If a file extends across volume boundaries before terminating, EOV labels are written, indicating that the file continues on another volume (see Figure 2-3).

CHAPTER 3

ACCESSING CONCEPTS

This chapter describes access checks, types of access, conditions that occur when files are read and written, control functions, and high-level language support for ANSI-labeled volumes on a VAX/VMS system.

3.1 ACCESSING EXISTING FILES

To access a particular file on a mounted volume, you must specify a file name. You can specify the file name explicitly or you can use wild card characters to request a set of files.

3.1.1 File Name Identification

When you specify a VAX/VMS or ANSI file name, the MTAACP will compare the file name with the file header labels of each file until it finds a match in the file identifier field of the HDR1 label.

If you supply a version number in the file name, it is compared with the generation number and generation-version number fields in the HDR1 label. If you do not specify a version number, the MTAACP neither defaults a version number nor checks the generation number and generation-version number fields. The MTAACP selects the first file on the tape whose file name in the file identifier field matches the specified file name.

Neither the directory nor the latest version number concept is supported for magnetic tape volumes. VAX/VMS does not search for the latest version of a specified file or list the latest version of a file; the MTAACP cannot increment version numbers of files written to magnetic tape. Therefore, two or more files in the same volume set can have the same file name and version number.

Because the MTAACP selects the first matching file name and version number (if specified), the position of the magnetic tape within the volume set determines which file is returned on a search operation. A search operation begins at the current position, so you may want to rewind the volume set before accessing a file.

If the requested file is not found on the current volume, the remaining volumes in the volume set are searched sequentially, according to their relative volume numbers, until either a file name match occurs or the entire volume set is searched.

If a file name match occurs, the internal file identification number is constructed from the file section number, file-sequence number, and

relative volume number. Although you can access a disk file by its file identification number, you cannot access a magnetic tape file this way.

After the file name comparison yields a match, the MTAACP then checks the accessibility field in the header to determine whether it contains a space or some other character. If it contains a space, you can access the file, and the MTAACP positions the tape at the beginning of the file section.

If the accessibility field contains any character other than a space, you cannot access the file unless you specified the /OVERRIDE=ACCESSIBILITY qualifier at mount time.

If the file is accessed for a write operation, VAX/VMS also checks the expiration date field in the HDR1 label.

3.1.2 Wild Card Character Identification

The asterisk (*) and percent sign (%) are the only valid wild card characters for magnetic tape volumes. The asterisk matches file specifications by field or portion of a field. The percent sign matches any character in a file specification only by character positions within a field.

VAX/VMS and ANSI file names support these wild card characters differently. With VAX/VMS file names, you can specify the asterisk and the percent sign anywhere in the file name and file type field to match file name specifications by field or character position within a field. You cannot use the percent sign in the version number field, only the asterisk.

With ANSI file names, you cannot use the percent sign. A single asterisk in a field is the only valid wild card character for ANSI file names.

Before specifying wild card characters to search a volume set for a set of files, you may want to rewind the volume. When you search a volume, the search begins at the first file that VAX/VMS accesses. The search for a matching file and version number (if specified) continues at the beginning of the header label set of the next file. The search ends when the magnetic tape is positioned at the file where the search began.

Examples

1.
```
$ DIRECTORY MFA1:*.*;*
```

   This command instructs VAX/VMS to search a volume set. Because asterisks are used in the file specification and the asterisk is a valid wild card character for both ANSI and VAX/VMS file names, both VAX/VMS and ANSI file names will be returned.

2.
```
$ DIRECTORY MTA1:%*.*;*
$ DIRECTORY MTA0:*.%*;*
```

   In these two commands, the search can only match with VAX/VMS file names because the percent sign is not valid for ANSI file names. In the second example, the file type field must contain at least one character. Files with no file type will not be returned.

3. $ DIRECTORY MTA0:*.;*

   In this example, the DIRECTORY command instructs VAX/VMS to search for files with ANSI file names, as well as VAX/VMS file names that have a null file type.

3.2 ACCESSING AN ANSI-LABELED VOLUME OR FILE

When you access an ANSI-labeled volume or a file, VAX/VMS checks at the volume and/or file level to ensure that your process can access the volume or file. The level at which VAX/VMS checks access depends on the operation that you request and the type of access that the operation requires.

3.2.1 Levels of Access

When you access a volume or a file, VAX/VMS reads the volume and header labels to determine whether access to the volume or file is restricted. Which label will be read depends on the operation that is requested. For example, if you want to mount a volume, your process must have access to it. Thus, VAX/VMS will read the VOL1 label only. If you want to initialize a volume, however, you will overwrite existing files. Therefore, VAX/VMS will read both the VOL1 label and the HDR1 label of the first file on the volume to make certain that your process is not restricted from accessing the volume to overwrite files. Volume-level and file-level access checks are discussed below.

3.2.1.1 Volume-Level Access Checks - When you mount or initialize a volume, VAX/VMS reads the accessibility field in the VOL1 label. When this field contains any character other than a space, VAX/VMS will not allow you to access the volume, and your process will receive this error message: magnetic tape file access is nonblank. To override this protection, thus allowing you to access the volume, specify the /OVERRIDE=ACCESSIBILITY qualifier with either the MOUNT or INITIALIZE command (see Chapter 4).

When you initialize a volume, VAX/VMS checks the accessibility field in the HDR1 label of the first file on the volume. If this field contains any character other than a space, VAX/VMS will not allow your process to access the volume and your process receives the above error message. To override this protection, specify the /OVERRIDE=ACCESSIBILITY qualifier with the INITIALIZE command.

VAX/VMS also checks the expiration date field in the HDR1 label of the first file on the volume when you initialize a volume. This date specifies when the volume and the file will expire. If the expiration date has not been reached, VAX/VMS will not allow your process to initialize the volume, and your process will receive this error message: file is not expired. To override this protection, specify the /OVERRIDE=EXPIRATION qualifier with the INITIALIZE command.

If the volume is a VAX/VMS ANSI-labeled volume, VAX/VMS also checks the owner identifier field in the VOL1 label and interprets the protection code and owner UIC to confirm that your process can access the volume. You can override protection by specifying the /PROTECTION=code qualifier when you mount or initialize a volume. The protection code that you specify overrides the protection written on the volume.

You must have VOLPRO privilege to specify any of these qualifiers for the MOUNT and INITIALIZE commands if all of the following are true:

- The volume is a VAX/VMS ANSI-labeled volume.
- You are not the volume owner or a SYSTEM user (see Section 2.5.3).
- The protection on the volume restricts your process from accessing the volume (see Table 2-2).

3.2.1.2 File-Level Access Checks - When you access a file, the MTAACP reads the accessibility and expiration date fields in the HDR1 label. If the accessibility field contains any character other than a space, the MTAACP does not allow you to access the file; your process receives this error message: magnetic tape file access is nonblank. To inhibit this check, specify the /OVERRIDE=ACCESSIBILITY qualifier with the MOUNT command when you mount a volume.

By default, VAX/VMS writes a space to the HDR1 accessibility field when you create a file. However, you can use VAX-11 RMS to set this field by specifying an ASCII "a" character in the MTACC field of the protection extended attribute block (XABPRO). (Refer to the VAX-11 Record Management Services Reference Manual for further information.) To access a file after you have set the accessibility field and closed the file, you must dismount the volume and remount it, specifying the /OVERRIDE=ACCESSIBILITY qualifier, if you did not do so when you mounted the volume.

The expiration date field can prevent you from overwriting or appending to a file. If the date written in the expiration date field has not been reached, the file has not expired. You cannot overwrite an unexpired file unless you specify the/OVERRIDE=EXPIRATION qualifier when you mount the volume.

If you do not specify the /OVERRIDE=EXPIRATION qualifier, the MTAACP checks the expiration date field for the file before it allows you to write to a file. When more than one file will be overwritten, the MTAACP also checks the expiration date of the file immediately following the file that will be overwritten. For example, before you append to a file, the MTAACP checks the expiration dates of both the file to be appended to and the file immediately following the file to be appended. If the expiration date of either file has not been reached, the MTAACP does not allow you to append to the file.

3.2.2 Types of Access

When you update, append, or create a file, you are requesting the MTAACP to access the file and read and/or write to it in a particular manner. The manner in which the file is accessed to perform the operation is called the access type. Read and/or write privilege is required, depending on the type of access required. The type of access also determines where the tape is positioned within a file. The following sections describe various access types.

3.2.2.1 Read Access - When a file is accessed for a read operation, the tape is positioned at the beginning of the file section after the file header labels.

3.2.2.2 File Attributes Access - When a file is accessed for reading only the attributes in the header labels, rather than the data in the file section, the MTAACP returns the VAX-11 RMS attributes to your process. For example, when you specify the DIRECTORY/FULL command for a volume, file, or list of files, the MTAACP selects the file identifiers from the HDR1 labels, returns the file attributes to your process, and positions the magnetic tape after the header labels of the last file accessed.

3.2.2.3 Write Access - When a file is accessed for a write operation, one of the following operations is actually being performed--append, update, or create. The position of the tape depends on which one of the three write operations you request. Each operation is described in a separate section below.

- Append Access

  When a file is accessed for an append operation, the tape is positioned at the end of the file section before the tape mark that precedes the trailer labels. After the file is appended and closed, all files beyond the appended file are lost. When the positioning is complete, the processing is handled as if the file had been created as described in Section 3.4.1.

- Update Access

  When a file is accessed for an update operation, the tape is positioned at the beginning of the file section after the header labels. After the file is written and closed, all files beyond the updated file are lost. The processing is handled as if the file had been created (see Section 3.4.1).

- Create If

  When you open a file, you can specify the Create If function with either VAX-11 RMS or $QIO calls. When you use this option and specify a file that exists, the file will be opened; if the file does not exist, it will be created and written following the last file on the last volume in the volume set. To specify this option with VAX-11 RMS, refer to the create if (CIF) file-processing option (FOP) for the file access block (FAB) in the VAX-11 Record Management Services Reference Manual.

3.3 READING FILES

The following sections discuss the conditions that can occur when VAX-11 RMS reads the data in the file section of a file.

3.3.1 End-of-Volume or End-of-File on Read

When the magnetic tape driver encounters a tape mark when reading data in a file, it returns an end-of-file status message to the MTAACP. The MTAACP does not return this end-of-file message to your process until it determines whether the end-of-file or the end-of-volume has been reached.

To determine which status exists, the MTAACP reads the first trailer label. If character positions 1 through 3 contain EOF, end-of-file has been found; if these character positions contain EOV, end-of-volume has been found.

In both cases, the block count in the EOF1 or EOV1 trailer label is compared to the accumulated block count to determine whether all the blocks in the current file section have been read. If the block count in the trailer label does not equal the accumulated block count, the compare operation fails; the warning status message BLOCK COUNT ERROR is returned to your process. If the compare operation succeeds and EOF was found, the informational status message END OF FILE is returned to your process. If the compare operation succeeds and EOV is found, the MTAACP performs a volume switch and reissues your read requests to the continuation volume. The MTAACP does not return a status message to your process. Therefore, this operation is transparent.

3.3.2 Volume Switch on Read

The MTAACP initiates a volume switch during a read operation when the trailer labels of the last file on a volume are EOV labels. Before switching volumes, the MTAACP checks the drive(s) associated with the volume set at mount time to determine whether the continuation volume is mounted.

If the continuation volume is mounted, the MTAACP switches volumes by reissuing all pending read requests to the continuation volume. If the volume is not mounted, the MTAACP selects the drive on which to mount the volume; it then sends a message through the operator's communication facility (OPCOM) of VAX/VMS to the operator's console to request the operator to mount the volume. The MTAACP messages and operator's responses are transparent to your process. However, if the operator is unavailable or cannot mount a volume immediately, you can experience a slight delay in system response until the volume is mounted.

If the operator is unavailable, you can load the continuation volume and mount it by specifying the REPLY command (see Section 3.3.2.1) from the operator's console to respond to any MTAACP requests for your own process. After the volume is mounted, the MTAACP checks the HDR1 label of the first file on the volume to ensure that the file-set identifier, file section number, and file sequence number match those it is seeking. If the correct volume is mounted, the MTAACP then reissues any pending read requests to the volume.

Although the MTAACP reads the header labels on the continuation volume, it does not return the information to your process. The file characteristics established when the file was first created apply to all subsequent file sections.

All MTAACP mount requests are sent to the operator's console; the MTAACP sends no message to your process. MTAACP mount requests and appropriate operator REPLY command responses are presented below.

3.3.2.1 MTAACP Mount Requests and REPLY Command Responses - A read operation is the only operation in which the volume identifier of the continuation volume need not be specified in either the MTAACP mount request or the REPLY command. There are two reasons why. First, the MTAACP request to mount a continuation volume always will be for an

existing volume in the volume set; a new volume will not be created. Second, the MTAACP reads the HDR1 label of the first file on the continuation volume to ensure that the correct volume is mounted and the correct file is being read.

An MTAACP mount request always specifies the following identification information:

- %OPCOM prefix
- Date
- Time
- Identification number of the MTAACP request
- Your process name
- Relative volume number of the continuation volume
- Drive on which the volume should be mounted

The MTAACP mount request specifies the volume identifier of the continuation volume only if you have specified the volume identifier but did not load and mount the volume when you mounted the volume set.

After you load the continuation volume on the drive specified in the MTAACP mount request, you (or the operator) mount the volume by issuing the REPLY/TO command, specifying the identification number in the MTAACP mount request; the volume identifier is optional.

When a volume identifier is not specified in either the MTAACP mount request or the REPLY command, you cannot mount a continuation volume unless the volume allows your process to access it and to read existing files. Before it mounts a volume for which a volume identifier was not specified, the MTAACP performs the access checks described in Section 5.2.3. If the volume fails any of these access checks, the MTAACP will not mount the volume. If the volume does not restrict access, the MTAACP mounts the volume and reissues pending read requests to the continuation volume.

Examples

1. $
   %OPCOM, 22-FEB-1982 15:23:31.78, request 1, from user PLAW
   MOUNT relative volume 2 (DW0QT2) on MTA1:

   This MTAACP mount request includes a volume identifier. After loading the volume on the specified drive, you could use either of the REPLY commands below to respond to this request.

   ```
   $ REPLY/TO=1 "DW0QT2"

   $ REPLY/TO=1
   ```

   The first command specifies the volume identifier; the second does not.

2. $
```
%OPCOM, 23-FEB-1982 15:23:31.78, request 2, from user CRANE
MOUNT relative volume 2 () on MTA1:
```

This MTAACP mount request instructs the operator to mount the second volume in the set on drive MTA1. You could use either of the REPLY commands below to respond to this request.

```
$ REPLY/TO=2 "AU"

$ REPLY/TO=2
```

The first command specifies the volume identifier; the second does not.

3.3.3 Closing a File Accessed for Read

A file is closed either implicitly or explicitly. The file is closed implicitly when the driver encounters a tape mark while a file is being read. The MTAACP then reads the trailer labels, closes the file, and positions the tape at the next file.

The file is closed explicitly when you deaccess the file before all the data in the file is read. The MTAACP will then close the file without reading the trailer labels, and the tape remains at the current position.

3.4 WRITING FILES

When files are written, the MTAACP performs access checks, writes labels, and, if necessary, switches volumes. The following sections describe conditions that can occur during write operations.

3.4.1 Creating a New File

When you create a new file using VAX-11 RMS or $QIO calls, you can specify a control function to indicate where the file is to be created. If you do not specify a location, the file is created after the last file on the last volume in the volume set. You can specify that the file will be created at the beginning of the volume set or at the next file position; if you do, however, all files that follow the created file will be lost after the file is created and closed.

If the new file will overwrite an existing file, the MTAACP checks the expiration date and accessibility fields of the existing file. If overwriting is allowed, the MTAACP overwrites the header label set of the existing file, creates the file section, writes the trailer labels, and writes two tape marks to denote the logical end-of-volume. All files following the newly created file are lost.

When you use $QIO calls to create a file, you should always specify the access function modifier (IO$M_ACCESS) in the create call (IO$_CREATE). If the access function modifier is not specified in the create call, the MTAACP is not informed that a file is being created and the trailer labels are not written. Therefore, if you unload or dismount the volume from the drive before you access the file, the file will not be properly closed and the tape will not have correct

ANSI labels. When you remount the volume and access this file, your process receives this error message: magnetic tape position is lost. For more information about this function modifier, refer to the VAX/VMS I/O User's Guide.

3.4.2 End-of-Tape on Write

When the driver encounters an EOT marker during a write operation, the MTAACP issues commands to the driver to write one or more tape marks (if necessary), the appropriate trailer labels (EOV or EOF labels), and two tape marks to indicate the logical end-of-volume.

The number of tape marks and the type of trailer labels that are written to the volume depend on what is being written when the driver encounters the EOT marker. If header labels are being written, the entire header label set plus two tape marks are written before a set of EOV trailer labels and two tape marks are written. If user data is being written, the current block of data and a single tape mark are written before a set of EOV trailer labels and two more tape marks are written. If a set of EOF trailer labels are being written, the entire set of labels and two tape marks are written.

When EOV labels are written to the volume, the MTAACP switches volumes and reissues pending write requests to the continuation volume. The operation is transparent to your process.

3.4.3 Volume Switch on Write

The MTAACP switches to a continuation volume during a write operation when the driver encounters an EOT marker but pending write requests exist. The MTAACP then closes out the current volume as described in Section 3.4.2. Before switching volumes, the MTAACP checks the drive associated with the volume set at mount time to determine whether the continuation volume is mounted. If the volume is mounted, the MTAACP reissues pending write requests to the continuation volume. If the volume is not mounted, the MTAACP selects the drive on which to mount the volume and then sends a message through the operator communication facilities (OPCOM) to the operator's console requesting the operator to mount a new volume on a specific drive.

The MTAACP messages and the operator's responses are transparent to your process. However, if the operator is unavailable or cannot mount the volume immediately, you can experience a slight delay in system response until the volume is mounted. If the operator is unavailable, you can load the continuation volume and mount it by specifying the REPLY command from the operator's console to respond to any MTAACP mount requests for your process (see Section 3.4.3.1).

After the volume is mounted, the MTAACP instructs the driver to write the volume label, header labels, and a tape mark to the volume if they have not been previously written. The MTAACP then reissues any pending write requests to the continuation volume.

All MTAACP mount requests are sent to the operator's console; the MTAACP sends no messages to your process. The MTAACP mount requests and appropriate operator REPLY command responses are presented below.

3.4.3.1 Responses to MTAACP Mount Requests - The MTAACP mount request always contains the following identification information:

- %OPCOM prefix
- Date
- Time
- Identification number of the MTAACP request
- Process name
- Relative volume number of the continuation volume
- Drive on which the volume should be mounted

The volume identifier also is specified in the request, but only if the number of volume identifiers that you specified for a given volume set at mount time exceeds the number of volumes that you mounted on drives. For example, if you explicitly created and mounted a volume set and specified the volume identifiers of all the volumes in the MOUNT command string but did not load all these volumes on drives, the volume identifier is specified in the MTAACP mount request.

You must specify the volume identifier of the continuation volume in either the MTAACP mount request or the REPLY command. Specifying a volume identifier ensures that the correct volume is mounted on the drive and links the continuation volume to the volume set.

To satisfy an MTAACP mount request, you must specify one of three qualifiers with the REPLY command: /TO, /INITIALIZE_TAPE, or /BLANK_TAPE. The volume identifier and the format written on the continuation volume determine which qualifier should be used.

For example, assume that the MTAACP issued the following mount request:

```
%OPCOM, 14-JUN-1982 15:23:31.78, request 3, from user PLAW
MOUNT new relative volume 2 (DW0QT2) on MTA1:
```

When the volume identifier written on the continuation volume is the same as that specified in the MTAACP mount request, the /TO qualifier should be used with the REPLY commands to satisfy the request, as shown in the two examples below.

```
$ REPLY/TO=3

$ REPLY/TO=3 "DW0QT2"
```

The first REPLY command does not specify the volume identifier; the second does. If you initialize and mount a volume set in which each volume has a unique accessibility character and you wish to retain that accessibility character, you must not specify the volume identifier. If you do specify the volume identifier, as in the second REPLY command, the accessibility character of the first volume in the set that was mounted will be written to the continuation volume, rather than the character specified at initialization.

When the volume identifier on the continuation volume does not match the one specified in the MTAACP mount request, the /INITIALIZE_TAPE qualifier should be specified with the REPLY command to reinitialize and mount the volume with the volume identifier.

The MTAACP will then perform access checks and initialize the volume as if the INITIALIZE command had been specified (see Section 4.2.3). However, if the volume fails any of the MTAACP access checks, the MTAACP cannot override them, and the volume cannot be initialized.

The first REPLY command does not specify the volume identifier; the second does.

```
$ REPLY/INITIALIZE_TAPE=3

$ REPLY/INITIALIZE_TAPE=3 "DW0QT2"
```

When a volume is unformatted, you must specify the /BLANK_TAPE qualifier to initialize the volume. VOLPRO privilege is required to avoid a runaway tape or timeout condition (see Section 4.2.5). Either of these next REPLY commands is valid. The first specifies a volume identifier; the second does not.

```
$ REPLY/BLANK_TAPE=3

$ REPLY/BLANK_TAPE=3 "DW0QT2"
```

The MTAACP can issue a mount request that does not specify a volume identifier, such as in the following request:

```
$
%OPCOM, 20-FEB-1982 15:23:31.78, request 4, from user PALP
MOUNT new relative volume 2 () on MTA1:
```

In this case, a volume identifier must be specified in the REPLY command, because it was not specified in the MTAACP mount request. The /TO qualifier should be used, as in:

```
$ REPLY/TO=4 "DW0QT2"
```

When the volume identifier written on the volume is not the volume identifier that you want to specify, you must specify REPLY/INITIALIZE_TAPE and a volume identifier that you select to initialize and mount the volume.

Before the MTAACP initializes the volume, the MTAACP performs access checks to ensure that your process can access and overwrite the volume. You cannot override access protection if your process is restricted from accessing the volume; the MTAACP will not initialize and/or mount the volume.

```
$ REPLY/INITIALIZE_TAPE=4 "DW0QT2"
```

When the volume is unformatted, you must specify the /BLANK_TAPE qualifier to initialize and mount the volume. VOLPRO privilege is required to avoid a runaway tape or timeout condition.

```
$ REPLY/BLANK_TAPE=4 "DW0QT2"
```

3.4.4 Closing a File Accessed for Write

To close a file, the MTAACP issues commands to the driver to write the EOF or EOV labels, which are followed by a double tape mark that indicates the logical end-of-volume. EOF labels are written when a file is closed unless the driver encounters an EOT marker.

3.5 CONTROL FUNCTIONS

A positioning operation, such as rewinding a file or volume, is performed by a control function that you specify through VAX-11 RMS or $QIO calls. When you specify a control function for a volume, files cannot currently be accessed. When you specify a control function for a file, the file must be currently accessed. The following sections describe control functions and their support by the MTAACP.

3.5.1 Rewinding a File

A rewind operation can be performed on an accessed file only. If data has been written to the file, the trailer labels will be written. If the first section of a file is on a previous volume, the MTAACP requests that a switch to the previous volume be made. If the previous volume is not online, the MTAACP sends a message through OPCOM to the operator's console requesting that the volume be mounted.

To use VAX-11 RMS to rewind a file, refer to the $REWIND macro in the VAX-11 Record Management Services Reference Manual. To rewind a file with $QIO calls, specify the IO$ACP_CONTROL function and the FIB$C_REWINDFIL value in the FIB$W_CNTRLFUNC field of the file information block (FIB). Refer to the VAX/VMS I/O User's Guide for further information.

3.5.2 Backspace Block and Forward-Space Block

A backspace or forward-space operation can be done on an accessed file only; labels cannot be accessed, and logical I/O can be requested only within a file section. If the file is accessed for a read operation, either a backspace or a forward-space operation is allowed. If the file is accessed for a write operation, only a backspace operation is allowed. If the previous operation was a write, the trailer labels are written before the backspace operation is completed. If necessary, the MTAACP switches volumes.

To use VAX-11 RMS to specify a backspace or forward-space operation, refer to the $SPACE macro in the VAX-11 Record Management Services Reference Manual. To specify either of these operations with $QIO calls, specify the IO$ACP_CONTROL function and the FIB$C_SPACE value in the FIB$W_CNTRLFUNC field of the file information block (FIB). For details, refer to the VAX/VMS I/O User's Guide.

3.5.3 Force Next Volume

The force-next-volume operation can be performed only on an accessed file. If the file is being written, the trailer labels will be written and the next volume will be mounted. Subsequent write requests are issued to the new volume. When a file is being read, the force-next-volume operation positions the magnetic tape at the trailer labels of the last file on the current volume. If there is another volume in the set, it is mounted and reading continues. If there is not another volume in the volume set, your process receives an informational end-of-file status message.

To specify a force-next-volume operation with VAX-11 RMS, refer to the $NXTVOL macro in the VAX-11 Record Management Services Reference Manual. To specify this operation with $QIO calls, specify the

IO$ACP_CONTROL function and the FIB$C_NEXTVOL value in the FIB$W_CNTRLFUNC field of the file information block (FIB). For details, refer to the VAX/VMS I/O User's Guide.

3.5.4 Position to End

The position-to-end operation can be performed only when no file is being accessed. In anticipation of the creation of a new file, the MTAACP positions the tape after the trailer labels of the last file on the last volume of the volume set. This positioning is the default on a VAX-11 RMS $CREATE operation. To specify a position-to-end operation with $QIO calls, specify the IO$ACP_CONTROL function and the FIB$C_POSEND value in the FIB$W_CNTRLFUNC field of the file information block (FIB). Refer to the VAX/VMS I/O User's Guide for further information.

3.5.5 Rewind Volume Set

The rewind-volume-set operation rewinds all volumes in a volume set. To specify this operation with $QIO calls, specify the IO$ACP_CONTROL function and the FIB$C_REWINDVOL value in the FIB$W_CNTRLFUNC field of the file information block (FIB). Refer to the VAX/VMS I/O User's Guide for further information.

3.6 HIGH-LEVEL LANGUAGE SUPPORT

Not all high-level languages supported by VAX/VMS can directly support or access all of the ANSI label facilities. However, most of the high-level languages allow you to call MACRO subroutines, and you can access all of the label facilities through MACRO.

For example, neither FORTRAN nor BASIC provides a mechanism that allows you to set the expiration date in the HDR1 label. However, both languages support USEROPEN keywords, which allow you to call MACRO subroutines to access the label facility. Figure 3-1 is a MACRO routine named SET_EXDATE that sets the expiration date in the HDR1 label of a file. Figure 3-2 is a FORTRAN program that calls SET_EXDATE, and Figure 3-3 is a BASIC program that also calls SET_EXDATE.

Example 3-1: MACRO Routine Sets Expiration Date

```
.TITLE   SET_EXDATE      ! MACRO routine that sets the expiration date

;++
; The USEROPEN keyword in the OPEN statement of the following FORTRAN
; and BASIC programs calls this SET_EXDATE MACRO routine from the
; Run-Time Library during execution of the OPEN statement.  The COMMON
; area EXP_DATE is input implicitly and contains the expiration date
; and time in a 64-bit, binary, absolute (if positive) or delta (if
; negative) value.  See the VAX/VMS System Services Reference Manual
; for details.
;--
```

Example 3-1 (Cont.): MACRO Routine Sets Expiration Date

```
;+
; Declare the PSECT which is the common area EXP_DATE.
;-

          .PSECT  EXPDATE RD,WRT,NOEXE,GBL,OVR,SHR,PIC,LONG
EXP_DATE:
          .BLKQ                          ;  64-bit binary date/time value
;+
; Set PSECT to one for executable code.
;-

          .PSECT  $CODE RD,NOWRT,EXE,LCL,CON,SHR,PIC
;+
; Define FAB and XAB symbols for later use.
;-

$FABDEF
$XABDEF
          .ENTRY  SET_EXDATE, M<R2,R3,R4,R5> ;  Procedure entry mask

FAB = 4
RAB = 8                                  ;  First argument is address of FAB
                                         ;  Second argument is address of RAB

          SUBL2   #XAB$C_DATLEN, SP              ;  Create DAT XAB on stack
          MOVC5   #0, (SP), #0, #XAB$C_DATLEN, (SP) ;  Clear XAB
          MOVB    #XAB$C_DAT, XAB$B_COD(SP)      ;  Initialize XAB
          MOVB    #XAB$C_DATLEN, XAB$B_BLN(SP)

          MOVQ    L^EXP_DATE, XAB$Q_EDT(SP)      ;  Set expiration date/time
          MOVL    FAB(AP), R0                    ;  Get FAB address
          MOVL    FAB$L_XAB(R0), XAB$L_NXT(SP)   ;  Link in XAB
          MOVL    SP, FAB$L_XAB(R0)

          $CREATE  FAB=@FAB(AP)                  ;  Create file
          BLBC    R0, 10$                        ;  If error, return
          $CONNECT  RAB=@RAB(AP)                 ;  Connect to record stream
10$:      RET                                    ;  Return with status in R0

          .END
```

Example 3-2: FORTRAN USEROPEN Keyword Calls MACRO Routine

```
                    .Title FOR_SET_EXDATE

C This FORTRAN program calls a MACRO routine that sets the expiration
C date in a HDR1 label when you create an ANSI file.  The COMMON area
C EXP_DATE contains the expiration date and time which consists of a
C 64-bit, binary, absolute (if positive) or delta (if negative) value.
C The MTAACP receives this value, truncates it, and writes only the
C date in the expiration date field of the HDR1 label.  The USEROPEN
C keyword in the OPEN statement calls the SET_EXDATE MACRO routine to
C set the expiration date in the HDR1 label.
```

Example 3-2 (Cont.): FORTRAN USEROPEN Keyword Calls MACRO Routine

```
        INTEGER*4 EXPDATE(2)            !  Expiration date/time
        COMMON /EXPDATE/ EXPDATE
        INTEGER*4 SYS$BINTIM            !  $BINTIM system service
        EXTERNAL SETEDT                 !  USEROPEN procedure

C The $BINTIM system service converts the expiration date and time
C from an ASCII string to a VAX/VMS 64-bit, binary, absolute or
C delta value.  See the VAX/VMS System Services Reference Manual
C for more details.

        ISTATUS = SYS$BINTIM ('1-JAN-1987',EXP_DATE)

C Check for error return from SYS$BINTIM.

        IF (.NOT. ISTATUS) CALL LIB$STOP (%VAL(ISTATUS))

C Open file;  USEROPEN keyword calls the SET_EXDATE MACRO routine.

        OPEN (UNIT=1,FILE='TAPEFILE',STATUS='NEW',
        1  USEROPEN=SET_EXDATE)

C .
C . (File processing occurs here)
C .
        CLOSE (UNIT=1)
        END
```

Example 3-3: BASIC USEROPEN Keyword Calls MACRO Program

```
                        .Title BAS_SET_EXDATE


0100 !  This BASIC program calls the MACRO routine SET_EXDATE in the
0200 !  example above to set the expiration date in the HDR1 label
0300 !  when you create a file.  Converted to a 64-bit, binary,
0400 !  absolute (if positive) or delta (if negative) value, the
0500 !  expiration date and time are stored in the COMMON area EXP_DATE.
0600 !  The USEROPEN keyword in the OPEN statement calls the SET_EXDATE
0700 !  MACRO routine to set the expiration date in the HDR1 label.
0800                                                          :
0900 COMMON (EXP_DATE) EDT%(1%)     ! Expiration date/time    .
1000 EXTERNAL INTEGER FUNCTION SYS$BINTIM ! $BINTIM system service
1100 EXTERNAL INTEGER FUNCTION SET_EXDATE ! USEROPEN procedure
1200 !
1300 !  The $BINTIM system service converts the expiration date and
1400 !  time from an ASCII string to a 64-bit, binary, absolute or
1500 !  delta value. See the VAX/VMS System Services Reference Manual
1600 !  for more details.
1700 !
1800
1900 ISTATUS% = SYS$BINTIM ("1-JAN-1987",EDT%(0%) BY REF)
2000
2100 !  Check for error return from SYS$BINTIM.
2200
2300 IF (ISTATUS% AND 1%) = 0% THEN &
     CALL LIB$STOP (ISTATUS% BY VALUE)
2400
```

Example 3-3 (Cont.): BASIC USEROPEN Keyword Calls MACRO Program

```
2500 !  Open file; USEROPEN keyword calls the SET_EXDATE routine.
2600
2700 OPEN 'TAPEFILE' FOR OUTPUT AS FILE #1%, &
     USEROPEN SET_EXDATE
2800
2900 !   .
3000 !   . (File processing occurs here)
3100 !   .
3200
3300 CLOSE #1%
3400 END
```

CHAPTER 4

PROCEDURES FOR ANSI-LABELED VOLUMES

This chapter describes how to use ANSI-labeled magnetic tape volumes.

4.1 ALLOCATE

Use the ALLOCATE command to logically assign a magnetic tape drive exclusively to your process. You should make this assignment before you load a volume on a drive to ensure that another user does not preempt the drive while you load the volume. In addition, because magnetic tape volumes are nonshareable, you must allocate the drive from the same process from which you will access the volume or you will not be able to access the volume.

Format

```
ALLOCATE device-name[:] [logical-name]
```

Command Parameters

device-name[:]

Specifies the drive on which the volume will be loaded. The device name can be a physical, generic, or logical name. A physical device name consists of a device code, alphabetic controller designation, and a unit number. A generic device name consists only of the device code. A logical name must equate to a physical or generic name. Use of the optional colon is recommended.

[logical-name]

Specifies an optional logical name to be associated with the specified tape drive.

Prompts

Device:

Log_Name:

Each ALLOCATE command allocates only one device to a process. Except for a list of generic device names, the ALLOCATE command does not accept lists of device names in the command string. Although you can specify a list of generic devices, only one drive from the list of specified generic device types will be allocated. The first available device will be the one allocated.

Examples

1. $ ALLOCATE MTA1:
   _MTA1: allocated

   This command specifies a physical device name MTA1:. VAX/VMS informs you that MTA1 has been allocated.

2. $ ALLOCATE MF,MT,MS DRIVE
   _MTA0: allocated

   This example specifies a list of generic device names. At a minimum, a generic device name consists of the device code; a controller designation is optional. Only one of the specified generic devices is allocated. Each element in the list must represent a unique generic device type.

   VAX/VMS informs you that drive MTA0: has been allocated. Although not indicated in the message, VAX/VMS also assigns the logical name DRIVE to the drive MTA0:.

3. $ ALLOCATE DRIVE1 D1
   _DRIVE1 allocated

   This example specifies a logical name DRIVE1 as the device name. VAX/VMS informs you that DRIVE1 has been allocated. Although not indicated in the message, VAX/VMS also assigns the new logical name D1 to the drive DRIVE1.

4.2 INITIALIZE

Use the INITIALIZE command to encode the VAX/VMS ANSI-labeled format on a magnetic tape volume. INITIALIZE writes labels and an empty file on the volume in the following order:

1. A VOL1 label
2. A HDR1 label and a HDR2 label with the file sequence number set to 0
3. Two tape marks framing an empty file
4. The corresponding EOF1 and EOF2 labels
5. A double tape mark, specifying logical end-of-volume

The HDR3 label is not written when a volume is initialized because the VAX-11 RMS attributes are not needed for the empty file that is created. Appendix B describe the fields and initialization values for the volume, header, and trailer labels.

Format

INITIALIZE device-name[:] volume-label

Command Parameters

device-name[:]

Specifies the drive on which the volume will be loaded. The device name can be a physical or logical name. A physical device name consists of a device code, alphabetic controller

designation, and a unit number. A logical name must equate to a physical or generic name. Use of the optional colon is recommended.

volume-label

Specifies a label for the magnetic tape volume to be initialized. Section 2.5 describes volume labels.

Prompts

Device: device-name[:]

Label: volume-label

4.2.1 Format and Protection Qualifiers

The following qualifiers and options affect the format of volumes when they are used with the INITIALIZE command. Unless otherwise specified, you must have VOLPRO privilege to access a VAX/VMS ANSI-labeled volume when the protection code, if any, restricts your process from accessing the volume. For additional qualifiers, refer to the VAX/VMS Command Language User's Guide.

/DENSITY=n

Specifies the density in bytes per inch (bpi) at which the volume will be written. You can specify a density of 800, 1600, or 6250, if the tape drive supports the density. Table 1-1 lists the DIGITAL magnetic tape drives and the densities that they support.

When you do not specify the density for an unformatted volume, the density defaults to 1600 bpi. For all other volumes, density defaults to the density at which the first record on the volume is written, if the drive supports that density. If the drive does not support the density, VAX/VMS returns an error message indicating bad parity or the tape is not labeled.

/LABEL=VOLUME_ACCESSIBILITY:"single character"

Specifies the character written in the accessibility field of the VOL1 label. The quotation marks are part of the command; you must place them around the single accessibility character. This field is used for general volume protection.

You can specify any valid ASCII "a" character. However, if you specify any character other than a space, you must specify the /OVERRIDE=ACCESSIBILITY qualifier and option when you mount or reinitialize the volume on a VAX/VMS system.

/OVERRIDE=(option,[...])

Overrides fields in ANSI labels that prevent write access to the volume. You can specify one or both of the following:

EXPIRATION — Overrides the expiration date on a volume. The expiration date of the first file on the volume denotes the volume expiration date. Use this option when initializing an unformatted volume or an ANSI-labeled volume on which the first file has not expired.

ACCESSIBILITY Overrides any character in the accessibility fields of the VOL1 and HDR1 labels. Use this option when initializing an unformatted volume or an ANSI-labeled volume on which the volume and/or file accessibility field contains any character other than a space.

If you specify more than one option, separate them with commas and enclose the list in parentheses.

/OWNER_UIC=uic

Assigns the user identification code (UIC) that you specify to the volume and its files. Specify the UIC variable in the format:

[g,m]

g is an octal number in the range 0 through 377 and denotes the group number.

m is an octal number in the range 0 through 377 and denotes the member number.

Either square ([]) or angle (<>) brackets are required in the UIC specification.

Use this qualifier to specify a UIC other than the default, which is the UIC of the current process. You must specify the /PROTECTION qualifier with the /OWNER_UIC qualifier or VAX/VMS writes no UIC to the volume.

/PROTECTION=code

Specifies the volume access protection code and writes a UIC on the volume. The volume access protection code controls who can access the volume and read and/or write files. Only read and write access are relevant to ANSI-labeled volumes; execute and delete access are not applicable. If you do not specify a protection code, all users have read and write access and no UIC is written on the volume, even if you specify the /OWNER_UIC qualifier. THe system and the owner are always given both read and write access regardless of the protection code that you specify. Table 2-2 lists the protection codes applicable to VAX/VMS ANSI-labeled volumes.

4.2.2 Access Checks Performed by INITIALIZE

Before it initializes a volume, INITIALIZE determines whether the format of the volume is ANSI-labeled. It also checks the accessibility fields in the VOL1 and HDR1 labels to see if access has been protected.

If the volume is ANSI-labeled, INITIALIZE checks the protection code and UIC in the owner identifier field of the VOL1 label to see whether your process can access the volume.

INITIALIZE also checks the expiration date in the HDR1 label of the first file on the volume; if that date has been reached, the file is expired.

If either of the accessibility fields contains any character other than a space or the first file on the volume has not expired, you must specify the /OVERRIDE qualifier with one or both the ACCESSIBILITY and the EXPIRATION options.

If the volume is a VAX/VMS ANSI-labeled volume containing volume protection that restricts your process from accessing the volume, you must have VOLPRO privilege to access the volume. For example, when the access protection restricts access to group users, your UIC group number must match the group number of the UIC written on the volume. If they do not match, you cannot access and initialize the volume unless you have VOLPRO privilege.

Thus, when you are neither the volume owner nor a system user and the access protection and UIC encoded on a volume restrict access, you must have VOLPRO privilege to override protection, access the volume, and specify the /PROTECTION and/or the /OVERRIDE qualifiers.

Examples

1. Specify the ALLOCATE command as shown below or described in Section 4.1.1 before you load the volume. Allocating a drive prior to loading the volume ensures that another user cannot preempt the drive while you load the volume. You must allocate the drive from the same process from which you mount the volume.

```
$ ALLOCATE MTA1:  TEST
_MTA1:  ALLOCATED
```

   VAX/VMS informs you that drive MTA1 is allocated. Although not indicated in the message, VAX/VMS assigns the logical name TEST to the drive MTA1.

2. Ensure the volume contains a write ring, if you want you to write to the volume, before you load the volume on the drive. A volume that does not contain a write ring is write-locked and cannot be written.

3. Press the on-line button if the drive does not go on line automatically. The indicator lamp should light informing you that the drive is on line.

4. Initialize (or reinitialize) the volume by using the INITIALIZE command.

   The following command would initialize the volume on drive MTA1: and would assign the volume label MYRAH to the volume.:

```
$ INITIALIZE MTA1:  MYRAH
```

   To initialize an unformatted volume, first make sure that you have VOLPRO privilege. Then, use a command such as:

```
$ INITIALIZE/OVERRIDE=(ACCESSIBILITY,EXPIRATION) VOLUME:  TESTER
```

   To initialize a non-VAX/VMS ANSI-labeled volume, use a command such as the following, which would initialize the

volume, set protection, write a specified UIC on the volume, set the density, override accessibility, and set the volume accessibility field:

```
$ INITIALIZE/PROTECTION=(G:RW,W:R)/OWNER_UIC=[012,074]/DENSITY=1600-
$_/OVERRIDE=(ACCESSIBILITY,EXPIRATION)/LABEL=VOLUME_ACCESSIBILITY:"B"-
$_MTA0:  MYRAH
```

4.3 MOUNT

Use the MOUNT command to check the volume identifier and accessibility fields in the VOL1 label before allowing your process access to an ANSI-labeled volume and its files. When the volume is a VAX/VMS ANSI-labeled volume, MOUNT also checks the owner identifier field in the VOL1 label.

Format

```
MOUNT device-name[:][,...] [volume-label [,...]]
[logical-name[:]]
```

Command Parameters

device-name[:][,...]

Specifies the drive on which the volume will be mounted. The device name can be a physical or logical name. A physical device name consists of a device code, alphabetic controller designation, and a unit number. A logical name must equate to a physical name. Use of the optional colon is recommended.

[volume-label[,...]]

Specifies a label for the magnetic tape volume to be initialized. The label is optional only if you specify the /OVERRIDE=IDENTIFICATION qualifier. Section 2.5 describes volume labels.

[logical-name[,...]]

Specifies an optional logical name to assign to the specified device.

Prompts

Device: device-name[:][,...]

Label: volume-label[,...]

Log_Name: logical-name[:]

The number of devices that you specify can be equal to, greater than, or less than the number of volume identifiers that you specify. If you specify more than one device or volume identifier, separate each element in each list with either a comma (,) or a plus sign (+).

The number of drives that you specify affects the action taken by the MTAACP when a continuation volume is required. For example, when the number of drives equals number of volume identifiers, the continuation volume is already mounted. The MTAACP switches volumes by issuing pending read or write requests to the next volume in the set.

When the number of drives is greater than the number of volumes, the MTAACP requests a continuation volume to be mounted on the first drive from the list that does not have a volume mounted.

When the number of specified drives is fewer, a volume must be dismounted before the system satisfies the MTAACP mount request. By specifying the drive on which the continuation volume is to be mounted, the MTAACP also determines which volume will be dismounted. Each time a continuation volume is requested, the MTAACP searches the list and extracts a drive that was specified at mount time.

For example, the first request for a continuation specifies that the volume will be mounted on the first drive in the list. The second request will specify the second drive. If another continuation volume is needed after the MTAACP has reached the end of the list, the MTAACP loops back to the beginning of the list and continues to select drives in a sequential fashion as continuation volumes are needed.

The MOUNT command implicitly allocates all drives specified in the original command string until you specify a DISMOUNT command or log out. However, if you explicitly allocate drives with the ALLOCATE command, you must specify the DEALLOCATE command.

If you specify more than one volume identifier, separate each with either a comma or plus sign. All volume identifiers specified in one MOUNT command must belong to the same file set, and the volume identifiers must be specified in ascending order according to their relative volume numbers.

If you specify more volume identifiers than drives, when the driver reaches the EOT marker while a file is being written or read, the MTAACP sends a message to the operator's console to request the operator to mount a continuation volume on a drive from the list specified at mount time (see Sections 3.3.2 and 3.4.3). This request is transparent to your process.

A logical name is optional. If you do not specify a logical name, VAX/VMS concatenates the string TAPE$ with the volume identifier of the first volume mounted, whether the volume set contains one or more volumes. If you specify trailing spaces in the logical name, VAX/VMS truncates them.

4.3.1 Format and Protection Qualifiers on the Mount Command

When used with the MOUNT command, the qualifiers described below affect the label format of a volume and/or the ancillary control process (ACP) used to process an ANSI-labeled volume. Unless otherwise noted, you must have VOLPRO privilege to use any of these qualifiers when the volume is a VAX/VMS ANSI-labeled volume containing protection that restricts your process from accessing the volume. Refer to the VAX/VMS Command Language User's Guide for additional qualifiers.

/BLOCKSIZE=n

> Specifies the default block size. Privilege is not required to use this qualifier. The range of valid values for n varies and depends on the density of the volume, whether the data is for input or output, and whether the operation uses VAX-11 RMS. For example, when the density is either 1600 or 6250 bpi, the block size on input can range from 1 to 65,532 bytes for VAX-11 RMS operations. However, if the density is 800 bpi, the minimum block size on input and output is 14 bytes for VAX-11 RMS and non-VAX-11 RMS operations.

For output, the minimum block size for VAX-11 RMS operations is 14 bytes for all densities.

The block size on input and output for non-VAX-11 RMS operations ranges from 1 to 65,534 bytes when the density is 1600 or 6250 bpi. By default, VAX/VMS writes 2048-byte blocks, which conform to the ANSI standard. Although VAX/VMS allows you to specify a block size larger than 2048 bytes, a larger block size does not conform to ANSI standards.

You must specify /BLOCKSIZE when you are mounting volumes that do not have HDR2 labels or support a block size smaller than 2048 bytes. For example, you must specify /BLOCKSIZE=512 to mount a RT-11 volume.

You must also specify /BLOCKSIZE when the volumes contain blocks that exceed the 2048-byte default block size. In this case, specify the size of the largest block for the block size.

/LABEL

Indicates that the volume contains standard ANSI labels. This is the default. Privilege is not required to use this qualifier. To specify foreign volumes, refer to Chapter 5.

/OVERRIDE=(option[,...])

Inhibits one or more of the (see Section 4.3.3) access checks performed by the Mount program and the MTAACP. The options are:

ACCESSIBILITY	Overrides any character in the accessibility field of the VOL1 and HDR1 labels that inhibits access. Use this qualifier at mount to override the accessibilty field in the VOL1 or a HDR1 label that contains any character other than a space.
EXPIRATION	Overrides the expiration dates of a volume and its files. Use this qualifier when the expiration date (in the HDR1 label) of any file that you want to overwrite has not been reached.
IDENTIFICATION	Overrides the volume identifier in the VOL1 label. Use this qualifier to mount a volume for which you do not specify the volume identifier. Only the volume identifier field will be overridden, volume protection, if any, is preserved.
SETID	Prevents the Mount program from checking the file-set identifier in the HDR1 label of the first file on a continuation volume. Use this qualifier only for ANSI-labeled volumes on which the file-set identifier of the first file on a continuation volume differs from the file-set identifier of the first file of the first volume that was mounted.

/OWNER_UIC=uic

Overrides the UIC written in the owner identifier field of the VOL1 label and assigns the UIC that you specify to the volume while it is mounted. For volume sets, in which a continuation volume is written, the UIC specified at mount time will be written to the volume only if the /PROTECTION qualifier was

specified either at mount time or when the volume had been initialized. Specify the UIC variable in the format:

[g,m]

g is an octal number in the range 0 through 377 and denotes the group number.

m is an octal number in the range 0 through 377 and denotes the member number.

Either square ([]) or angle (<>) brackets are required in the UIC specification.

/PROCESSOR=option

Requests the MOUNT command to associate an ancillary control process (ACP) to interpret the logical file structure of the volume. You must have the operator user privilege (OPER) to use the /PROCESSOR qualifier. This qualifier overrides the default manner in which VAX/VMS associates ACPs with devices. You can specify any one of the following options:

UNIQUE	Create a new process to execute a copy of the default ACP image for the specified device type or controller.
SAME:device	Use the same ACP process currently being used by the device specified.
partial file-spec	Create a new process to execute the ACP image specified by the file-spec (for example, a modified or a user-written ACP). Specify the file name, type and version number. The file must reside in SYS$SYSTEM. Node, disk, and directory specifications or wild card characters are not allowed in the partial file specification parameter.

/PROTECTION=code

Overrides the access protection written in the owner identifier field of the VOL1 label and assigns the protection code that you specify to the volume while it is mounted. For volume sets in which a continuation volume is written, the protection code that you specify will be written to the continuation volume. By default, your process UIC also will be written to the continuation volume unless you explicitly specify an alternate UIC with the /OWNER_UIC qualifier described above.

Valid protection codes include read and write access for group and world users; execute and delete access are not applicable to magnetic tape volumes. System users and the volume owner always have read and write access to magnetic tape volumes regardless of the protection code that you specify. Section 2.5.3 describes access and protection codes. Table 2-2 lists the protection codes.

/NOHDR3

Controls whether HDR3 labels are written to a volume. Privilege is not required for this qualifier. The default is /HDR3 which allows HDR3 labels to be written to a volume. Use the /NOHDR3 qualifier when writing to volumes that will be read by a system, such as the RT-11 system, which does not process HDR3 labels correctly.

/RECORDSIZE=n

Specifies the number of bytes in each record. This qualifier does not require privilege. Use this qualifier when you mount volumes without HDR2 labels (such as RT-11 volumes) to provide VAX-11 RMS with the size of fixed-length records or the maximum size of variable-length records.

The record size must be less than or equal to the specified or default block size. Refer to the /BLOCKSIZE qualifier (described above) for details. VAX/VMS will not write records smaller than 14 bytes on output. However, the Convert Utility, described in the VAX-11 Record Management Services Utilities Reference Manual, allows you to pad and extend the size of records up to and greater than the 14-byte minimum record size output by VAX/VMS.

4.3.2 Access Checks Performed by MOUNT

The MOUNT command checks whether the volume has either a VAX/VMS or non-VAX/VMS ANSI-labeled format. If the format is ANSI labeled, MOUNT checks the following:

- The volume identifier field
- The accessibility field in the VOL1 label contains a space
- If the volume is a VAX/VMS ANSI-labeled volume, the MTAACP reads the protection code and UIC in the owner identifier field of the VOL1 label to ensure your process can access the volume

Similar to the INITIALIZE command, the MOUNT command also supports an /OVERRIDE qualifier and options (see Section 4.3.1) to override access restrictions.

4.3.3 Mounting a Volume

This procedure describes how to mount an ANSI-labeled volume.

1. Specify the ALLOCATE command as shown below or described in Section 4.1 before you load the volume. Allocating a drive prior to loading the volume ensures that another user cannot preempt the drive while you load the volume. You must allocate the drive from the same process from which you mount the volume.

```
$ ALLOCATE MTA1:  TEST
_MTA1:  ALLOCATED
```

   VAX/VMS informs you that drive MTA1 is allocated. Although not indicated in the message, VAX/VMS assigns the logical name TEST to the drive MTA1.

2. Make sure that the volume contains a write ring before you load the volume on the drive. A volume that does not contain a write ring is write-locked and cannot be written.

3. Press the on-line button if the drive does not go on line automatically. The indicator lamp should light informing you that the drive is on line.

4. Specify the MOUNT command string, including the device name and volume identifier as shown below (specifying a logical name is optional):

```
$ MOUNT MTA1:  GEORGE MYVOL
%MOUNT-I-MOUNTED, GEORGE mounted on MTA1:
```

VAX/VMS informs you that the volume named GEORGE is mounted on the drive MTA1. Although not indicated in the message, VAX/VMS also assigns the logical name MYVOL to the volume GEORGE mounted on MTA1.

4.3.4 Mounting a Volume Set

To mount a volume set follow the procedure below.

1. For each drive on which you will load and mount a volume, specify the ALLOCATE command as shown below or described in Section 4.1. In the two examples below, the first ALLOCATE command specifies the device with a physical device name, while the second specifies a generic device type.

```
$ ALLOCATE MTA0:
_MTA0:  allocated
$ ALLOCATE MT:
_MTA1:  allocated
```

VAX/VMS informs you which drive is allocated for each ALLOCATE command that you specify.

You need not allocate a drive for each volume in a volume set. The MTAACP will request volumes be switched to appropriate drives if a continuation volume is required.

2. Make sure that all the volumes in the set contain write rings if you want to write to any one of the volumes. If even one of the volumes in the set does not contain a write ring at mount time, all volumes in the set will be write locked and cannot be written.

3. Load the volumes on the drives that you have allocated and place the drives on-line.

4. Specify the MOUNT command string including the allocated drives, the volume identifiers of the volumes in the set, and a logical name. Volume identifiers must be listed according to their relative volume numbers.

```
$ MOUNT MTA0:,MTA1:  TAPE1,TAPE2,TAPE3 SKI
%MOUNT-I-MOUNTED, TAPE1 mounted on MTA0:
%MOUNT-I-MOUNTED, TAPE2 mounted on MTA1:
```

VAX/VMS informs you that the volumes were mounted on the specified drives. Although not indicated in the message, VAX/VMX also stores the volume identifier TAPE3 in the I/O data base and assigns the logical name SKI to the volume set.

VAX/VMS stores but cannot verify the volume identifiers of volumes that you specify but do not physically load on drives at mount time. VAX/VMS later verifies the volume identifiers of such volumes when the volumes are accessed as described in the next section.

4.3.5 Mounting a Continuation Volume

Regardless of the operation that you are performing, when a continuation volume is required, the MTAACP sends a message to the operator's console to request that a volume to be mounted. This request is transparent to your process.

When a continuation volume is mounted and the first operation performed on the volume is a read operation, the volume identifier is verified only when it is specified by you or the operator who mounts the volume. The MTAACP does not need to verify the volume identifier because the file section number, file-set identifier, and the file sequence number in the HDR1 label of the file on the continuation volume ensure that the correct volume is mounted.

When a continuation volume is mounted and the first operation performed on the volume is a write, the MTAACP verifies the volume identifier specified by you or the operator who mounts the volume (see Section 3.4.3.1).

The MTAACP does not check the volume identifier when the /INITIALIZE_TAPE or /BLANK_TAPE qualifier is specified with the REPLY command (see Section 3.4.3).

4.4 CREATING A VOLUME SET

If you do not create a volume set explicitly, VAX/VMS will create one if necessary. If you have not mounted a volume set and a continuation volume is required, satisfying an MTAACP request to mount the continuation volume implicitly creates a volume set. For example, while writing to the only volume that you mounted, the driver encounters the EOT marker during a write operation. The MTAACP sends a message to the operator's console requesting that another volume be mounted.

After you or the operator mounts the volume, the MTAACP writes the volume and header labels and then reissues the pending write requests to the continuation volume. The file-set identifier in the HDR1 label of all files written to the continuation volume will be the file-set identifier of the first file on the first volume. The file-set identifier for VAX/VMS volume sets is always that of the first file of the first volume that was mounted in the set.

To explicitly create a volume set, follow this procedure:

1. Allocate a drive on which you will load each volume.

```
$ ALLOCATE MTA0:
_MTA0:  allocated

$ ALLOCATE MTA1:
_MTA1:  allocated
```

2. Ensure that each volume contains a write ring. A volume that does not contain a write ring is write-locked and cannot be written. The volumes must not be write-locked because they will be written when they are initialized.

3. Load the volumes on the allocated drives and place the drives on-line.

4. Initialize the volumes. You should specify the density and the access protection in addition to the device name and volume identifier in the INITIALIZE commands, as in:

```
$ INITIALIZE/DENSITY=1600/PROTECTION=(G:RW) MTA0:  TAPE1
$ INITIALIZE/DENSITY=1600/PROTECTION=(G:RW) MTA1:  TAPE2
$ INITIALIZE/DENSITY=1600/PROTECTION=(G:RW) MTA1:  TAPE3
```

If the number of volumes is greater than the number of drives available to the volume set, as in the above example, you may want to use the REQUEST command to ask the operator to assist you in switching, initializing, or mounting the volumes. The VAX/VMS Command Language User's Guide describes this command in detail.

5. Mount the volumes. You should include the drives and volume identifier. Specifying a logical name for the volume set is optional.

```
$ MOUNT MTA0:,MTA1:  TAPE1,TAPE2,TAPE3 TEST
%MOUNT-I-MOUNTED, TAPE1 mounted on MTA0:
%MOUNT-I-MOUNTED, TAPE2 mounted on MTA1:
```

VAX/VMS not only confirms which volumes have been mounted but also indicates on which drive each volume has been mounted.

VAX/VMS mounts and verifies only the volumes that are physically loaded on the drives at mount time. However, the volume identifiers of additional volumes that you specify will not be verified until the volumes are accessed.

You can check the densities, volume labels, UICs, and relative volume numbers of the volumes that are mounted on drives. Specify the SHOW DEVICES/FULL command to do so. If you specify a generic device code for the magnetic tape drives, such as MT, information for all the drives of that type that are configured in the system will be displayed. To display information for a volume mounted on a specific drive, specify the physical device code, consisting of the generic device code, the controller designation, and the unit number followed by a colon. A list of devices cannot be specified with this command.

For example, the following SHOW DEVICES command uses the generic device code MT.

```
$ SHOW DEVICES/FULL MT

     Device   MTA0:      14-JUN-1982 10:19:18.60

     on line
     Mounted
     Error Logging
     Enabled
     Allocated

Error count                    0     Owner process id    00060016
Operations completed        1809     Owner process name  _TTH5:
Reference count                1     Default buffer size     2048
Density                1600 BPI

Volume label        TAPE1              Record size                  0
Owner UIC           [010,175]          Transaction count            1
Volume protection   3000               Mount count                  1
ACP process name    MTAOACP            Relative Volume No.          1

     Device   MTA1:      14-JUN-1982 10:19:18.60

     on line
     Mounted
     Error Logging
     Enabled
     Allocated

Error count                  136     Owner process id    00060016
Operations completed        4779     Owner process name  _TTH5:
Reference count                1     Default buffer size     2048
Density                1600 BPI

Volume label        TAPE2              Record size                  0
Owner UIC           [010,175]          Transaction count            1
Volume protection   3000               Mount count                  1
ACP process name    MTAOACP            Relative Volume No.          2
```

4.5 COPYING FILES

Use the COPY command to copy files to and from ANSI-labeled volumes. VAX/VMS supports sequential, relative, and indexed files, but only sequential files can be copied to ANSI-labeled volumes. The only valid record formats are variable-length and fixed-length records; spanned records are not supported by VAX/VMS.

Although VAX/VMS supports stream and VFC records, it encodes these records in a variable-length format on ANSI-labeled volumes. Non-VAX/VMS systems do not distinguish stream records from VFC records; instead, they interpret both as variable-length records. Therefore, neither stream nor VFC records should be created on volumes that will be used for information interchange to a non-VAX/VMS system.

When you copy files to magnetic tape volumes, make sure that the files are in order according to their expiration dates. The first file on the volume should expire last; subsequent files should have earlier expiration dates. If files are not copied with their expiration dates in descending order, files that expire later than preceding files can be overwritten.

When you copy files from disk to ANSI-labeled volumes, the following items are not preserved:

- Directory specifications
- Individual file protection
- User identification code (UIC)
- Creation and expiration times but the dates are preserved
- Revision and backup dates and times

The full set of ASCII "a" characters and the 17-character ANSI file name are supported only for ANSI-labeled volumes, not for disk volumes. Therefore, when you copy files with ANSI file names from tape to disk, specify a standard VAX/VMS file name for the output file name specification. If you do not specify a VAX/VMS file name on output, your process receives the following error message: error in file name. This means that the ANSI file name is not a valid file name specification.

Although the version number default is always 0, if you do not specify a version number, VAX/VMS interprets the default differently for disks and tapes. When you copy a file from a disk volume, the file with the highest version number will be copied if you do not specify a version number. When you copy a file from a tape volume, the first file that matches the file name specification will be copied.

In addition, file version numbers are not incremented when files are copied to tape volumes, even when a file with the same name and version number already exists on the volume. Therefore, two or more files with the same file name and version number can exist on the same volume set. Thus, the location of a file within a volume set determines which file VAX/VMS copies.

Wild card characters are supported on input and output. For details, refer to Section 3.1.2 and the COPY command in the VAX/VMS Command Language User's Guide.

Examples

1.
```
$ COPY/LOG MTA1:"%&*?!SKI! *** SEASON.DAT
%COPY-S-COPIED, MTA1:[]"%&*?!SKI! ***.;1 copied to WRKD:[MANUAL]SEASON.DAT;1 (120 records)
```

   When you specify the /LOG qualifier, VAX/VMS returns a message that confirms the file was copied as specified and informs you how many records were copied. The ANSI file %*?!SKI!#" (# means space) is copied to the file SEASON.DAT on the default disk and directory WRKD:[MANUAL]. The file could not have been copied to disk unless the new file name was specified. VAX/VMS provided defaults for segments of the file specification that were not specified.

2.
```
$ COPY/LOG FORTAP.DAT MTA1:"%&*?!SKI! ** "
%COPY-S-COPIED, WRKD:[MANUAL]FORTAP.DAT;1 copied to MTA1:[]"%&*?!SKI! ***.;0 (120 records
```

   In this command an ANSI file name was specified as the output file specification. Note that VAX/VMS truncates the trailing space in the file name %&*?!SKI!##"# (where # means space).

3.
```
$ COPY *%.JOU;* MTA1:*.*
%COPY-S-COPIED, WRKD:[MANUAL]C6.JOU;1 copied to MTA1:[]C6.JOU;1 (4 records)
```

In this example, all files with a 2-character file name and a file type of JOU will be copied to the volume MTA1 with the same file name and type as they had on the disk volume. Version numbers will not be incremented.

4.
```
$ COPY MTA0:*.* *
%COPY-S-COPIED, MTA0:[]TASTETEST.DAT;1 copied to WRKD:[FOOD]TASTETEST.DAT;1 (249 records)
%COPY-S-COPIED, MTA0:[]CUKES.DAT;1 copied to WRKD:[FOOD]CUKES.DAT;1 (148 records)
%COPY-S-NEWFILES, 2 files created
```

In this example, neither file on the volume had an ANSI file name. Therefore, both files were copied to the disk volume.

5.
```
$ COPY MTA1:*.* [source]
%COPY-S-COPIED, MTA1:[]TAPETEST.DAT;1 copied to WRKD:[SOURCE]TAPETEST.DAT;1 (10) records)
%COPY-E-OPENOUT, error opening WRKD:[SOURCE]'%%%()!SKI! ***.;1 as output
-RMS-F-FNM, error in file name
%COPY-W-NOTCOPIED, MTA1:[]'%%%()!SKI! ***.;1 not copied
%COPY-E-OPENOUT, error opening WRKD:[SOURCE]'SANFRAN%%%***.;1 as output
-RMS-F-FNM, error in file name
%COPY-W-NOTCOPIED, MTA1:[]'SANFRAN%%%***.;1 not copied
%COPY-S-COPIED, MTA1:[]C6.JOU;1 copied to WRKD:[SOURCE]C6.JOU;1 (4 records)
%COPY-S-NEWFILES, 2 files created
```

In the above example, the COPY command string specifies that all files on the volume mounted on drive MTA1 should be copied to the current default disk and directory WRKD:[MANUAL]. However, files with ANSI file names are not copied to the disk because an asterisk (*) was specified in the file name field; VAX/VMS returns an error message to the process.

4.6 DISMOUNTING VOLUMES

Use the DISMOUNT command to dismount a volume or volume set. When you specify the DISMOUNT command, a volume is unloaded from the drive unless you specify the /NOUNLOAD qualifier. Although the volume is logically dismounted from the drive, the /NOUNLOAD qualifier allows the volume to remain physically loaded on the drive and rewinds the volume to the BOT marker. If you dismount a single volume in a volume set, VAX/VMS dismounts all the volumes in the set. You can save time and eliminate unnecessary handling of a volume by using this qualifier when you will remount or reinitialize a volume that you had mounted.

A volume will be dismounted and unloaded automatically if the system fails or if you log out of the process from which you had mounted the volume. Usually, a volume and its data will not be corrupted when the volume is dismounted without using the DISMOUNT command. However, data will be corrupted if the system fails or you unload a volume that contains an open file for which trailer labels have not been written. When you remount the volume and access the file without trailer labels, your process will receive this error message: magnetic tape position lost. All files preceding the file without trailer labels can be accessed until you access the file without trailer labels.

Example

```
$ DISMOUNT/NOUNLOAD MTA1:
$
```

In this example, the volume will remain loaded on drive MTA1: after the volume is logically dismounted. No message is returned.

4.6.1 Deallocating Drives

Logging out of a process from which drives have been allocated will automatically deallocate all explicitly and implicitly allocated drives. When a drive was explicitly allocated with the ALLOCATE command, you should explicitly deallocate the drive by specifying the DEALLOCATE command. A complement to the ALLOCATE command, the DEALLOCATE command logically deassigns a drive from your process and returns it to the pool of devices.

Example

```
$ DEALLOCATE MFA1:
$
```

This DEALLOCATE command logically deassigns drive MFA1 from the your process. VAX/VMS returns to the dollar sign prompt.

4.7 INTERCHANGING ANSI-LABELED VOLUMES

Volumes containing the ANSI standard format can be used on various types of systems that support the ANSI format. However, systems can support or base their formats on any one of the four levels of the ANSI standard for magnetic tape labels and file structure for information interchange. Levels of the ANSI standard are upward compatible only; higher levels support components of lower levels but lower levels do not support components of the higher levels. For example, a system supporting Level 3 of the standard, inherently supports Levels 1 and 2 but does not necessarily support any components of Level 4. Some systems do not support all the components of a particular level, while other systems support optional components. Therefore, before you attempt to use an ANSI-labeled volume on another system, you should know the following:

- The density at which the volume was written
- The volume identifier
- The level of the ANSI standard supported by both systems
- The record format, record size, and block size for volumes that do not contain HDR2 labels. These volumes support only a fixed-length record format.

The following sections provide guidelines for interchanging volumes.

4.7.1 Interchanging Volumes to a VAX/VMS System

Except for a nonzero buffer offset field in HDR2 labels and User header (UHLa) and trailer (UTLa) labels, VAX/VMS supports Level 3 of

the ANSI standard for magnetic tape labels and file structure for information interchange. If the HDR2 label of a file contains a nonzero buffer offset, VAX/VMS interprets the buffer offset to be part of the data in the file section. Therefore, interpretation of the data is unpredictable and unreliable.

VAX/VMS ignores user header and trailer labels on input and does not write them on output. Because user header and trailer labels are system-defined and system-dependent, not supporting them does not prevent VAX/VMS from interpreting data in the file sections. With the exception of HDR3 header labels and EOF3 and EOV3 trailer labels on VAX/VMS ANSI-labeled volumes, non-VAX/VMS ANSI HDR3, EOF3, and EOV3 labels and optional ANSI volume, header, and trailer labels are ignored on input.

Using the mount procedure described in Section 4.3.4 and the information listed in Section 4.7, you can mount volumes written by systems supporting Levels 1 through 4 of the ANSI standard for magnetic tape labels and file structure for information interchange on a VAX/VMS system. Qualifiers allowing you to override volume access protection, specify an ACP, or define record and block sizes, are described in Section 4.3.2.

However, the files must contain zero buffer offsets and cannot contain spanned records for VAX/VMS to process the data correctly. Because VAX/VMS is based on Level 3 of the above ANSI standard, VAX/VMS does not support the spanned record format, a Level 4 component. You must use logical I/O to read spanned records. For details, refer to the VAX/VMS I/O User's Guide.

4.7.2 Creating a Volume for Interchange

This section describes how to create a volume on a VAX/VMS system for use on a recipient system that supports Level 3 or 4 of the ANSI standard.

The first step is to allocate a tape drive:

```
$ ALLOCATE MFA1:
```

Next, you initialize the volume, as in this command:

```
$ INITIALIZE/DENSITY=6250 MFA1:  MYTAPE
```

To ensure that the recipient system can access the volume, do not use the /PROTECTION or /LABEL=VOLUME_ACCESSIBILITY qualifiers with the INITIALIZE command.

Next, specify the MOUNT command including the drive, volume identifier, and logical name as shown below:

```
$ MOUNT MFA1:  MYTAPE MYVOL
```

If you will use the volume on a system that does not support the maximum 2048-byte block size or HDR2 labels, you should specify the block and record sizes using the /BLOCKSIZE and /RECORDSIZE qualifiers (See Section 4.3.1). For variable-length records, the record size should be at least four bytes smaller than the block size. Unless explicitly specified, VAX/VMS defaults the block size to 2048 bytes.

Use the COPY/LOG command shown below to copy files from a directory on disk to the magnetic tape volume. The wild card characters specified

on input request the highest version of all files from the specified directory be copied to the volume MYVOL. The logical name specified at mount time replaces the physical device specifications.

```
$ COPY/LOG [BOOK]*.* MYVOL:

%COPY-S-COPIED, DISK$WORK:[BOOK]APPENDIX.RNO;4 copied to MYVOL:APPENDIX.RNO;4 (12 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]CHAPTER.RNO;7 copied to MYVOL:CHAPTER.RNO;7 (7 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]COMMAND.RNO;1 copied to MYVOL:COMMAND.RNO;1 (82 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]COPYRT.RNO;10 copied to MYVOL:COPYRT.RNO;10 (44 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]DOCPLAN.RNO;1 copied to MYVOL:DOCPLAN.RNO;1 (186 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]ENTRY.RNO;3 copied to MYVOL:ENTRY.RNO;3 (4 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]EXAMPLE.RNO;2 copied to MYVOL:EXAMPLE.RNO;2 (9 records)
%COPY-S-COPIED, DISK$WORK:[BOOK]FIGURE.RNO;1 copied to MYVOL:FIGURE.RNO;1 (7 records)
%COPY-S-NEWFILES, 8 files created
```

The /LOG qualifier informs you which files were copied; returns the disk, directory, and file specifications denoting from and to where the files were copied; and tallies the number of records within each file and the number of new files that were created. Refer to the VAX/VMS Command Language User's Guide for more information on the COPY command.

Next, use the DISMOUNT command, including the physical or logical device name followed by a colon, to dismount the volume.

```
$ DISMOUNT MTA1:
```

Finally, use the DEALLOCATE command to logically deassign the drive from your process and return the device to the pool of free devices:

```
$ DEALLOCATE MTA1:
```

CHAPTER 5

PROCEDURES FOR FOREIGN VOLUMES

This chapter describes how to use foreign volumes on VAX/VMS. A foreign volume has a format other than ANSI-labeled. Because VAX/VMS has no way of knowing what format to expect from other systems, it does not provide an ancillary control process to interpret the logical format of any foreign volume and does not recognize foreign volumes as being file-structured. In addition, because other systems may not support the ANSI standards, VAX/VMS can create foreign volumes for use on those other systems.

With the exception of the File Transfer Utility (FLX), which interprets the format of and allows you to process DOS-11 volumes, VAX/VMS provides no utilities for interpreting the formats of and processing foreign volumes. Therefore, you must provide your own program to process a foreign volume.

The formats of foreign volumes are classified as either labeled or unlabeled. A foreign labeled volume contains non-ANSI standard labels that structure, identify, and describe the volume and its data. For example, a DOS-11 formatted volume is a foreign labeled volume. Foreign unlabeled volumes contain only data and do not have labels. User specified block and record sizes allow VAX/VMS to read and write data on foreign unlabeled volumes.

5.1 ALLOCATE

Use the ALLOCATE command to logically assign a magnetic tape drive exclusively to your process. You should make this assignment before you load a volume on a drive to ensure that another user does not preempt the drive while you load the volume. In addition, because magnetic tape volumes are nonshareable, you must allocate the drive from the same process from which you will access the volume or you will not be able to access the volume.

Format

```
ALLOCATE device-name[:] [logical-name]
```

Command Parameters

device-name[:]

Specifies the drive on which the volume will be loaded. The device name can be a physical, generic, or logical name. A physical device name consists of a device code, alphabetic controller designation, and a unit number. A generic device name

consists only of the device code. A logical name must equate to a physical or generic name. Use of the optional colon is recommended.

[logical-name]

Specifies an optional logical name to be associated with the specified tape drive.

Prompts

Device: device-name[:]

Log_Name: logical-name

Each ALLOCATE command allocates only one device to a process. Except for a list of generic device names, the ALLOCATE command does not accept lists of device names in the command string. Although you can specify a list of generic devices, only one drive from the list of specified generic device types will be allocated. The first available device will be the one allocated.

Examples

1. $ ALLOCATE MTA1:
   _MTA1: allocated

   This command specifies a physical device name MTA1:. VAX/VMS informs you that MTA1 has been allocated.

2. $ ALLOCATE MF,MT,MS DRIVE
   _MTA0: allocated

   This example specifies a list of generic device names. when you allocate a drive. At a minimum, a generic device name consists of the device code; a controller designation is optional. Only one of the specified generic devices is allocated. Each element in the list must represent a unique generic device type.

   VAX/VMS informs you that drive MTA0: has been allocated. Although not indicated in the message, VAX/VMS also assigns the logical name DRIVE to the drive MTA0:.

3. $ ALLOCATE DRIVE1 D1
   _DRIVE1 allocated

   This example specifies a logical name DRIVE1 as the device name. VAX/VMS informs you that DRIVE1 has been allocated. Although not indicated in the message, VAX/VMS also assigns the new logical name D1 to the drive DRIVE1.

5.2 MOUNT

When you mount a foreign volume, you must specify either the /FOREIGN or /NOLABEL qualifier. You can mount only one volume at a time. VAX/VMS does not support foreign volume sets.

Format

```
MOUNT /FOREIGN device-name[:] [volume-label] [logical-name]
      /NOLABEL
```

Command Parameters

device-name[:]

Specifies the drive on which the volume will be mounted. The device name can be a physical or logical name. A physical device name consists of a device code, alphabetic controller designation, and a unit number. A logical name must equate to a physical name. Use of the optional colon is recommended.

[volume-label]

Specify a placement holder such as a space enclosed in quotation marks (" "). VAX/VMS neither reads nor writes labels on a foreign volume.

[logical-name]

Specifies an optional logical name to assign to the specified device.

Prompts

Device: device-name[:]

Label: [volume-label]

Log_Name: [logical-name]

The MOUNT command implicitly allocates the drive specified in the original command string until you specify a DISMOUNT command or log out. However, if you explicitly allocate the drive with the ALLOCATE command, you must specify the DEALLOCATE command.

A logical name is optional. If you do not specify a logical name, VAX/VMS concatenates the string TAPE$ with the volume identifier. If you specify trailing spaces in the logical name, VAX/VMS truncates them.

5.2.1 Format and Protection Qualifiers on the MOUNT Command

The qualifiers below set format and protection characteristics for foreign volumes. Unless otherwise specified, no privilege is required to use these qualifiers.

/BLOCKSIZE=n

Specifies the default block size. Valid values for the variable n vary and depend on the density at which a volume is written, whether the data is for input or output, and whether VAX-11 RMS is used. For example, when the density is either 1600 or 6250 bpi, the block size on input can range from 1 to 65,532 bytes for VAX-11 RMS operations. However, when the density is 800 bpi, the minimum block size on input or output is 14 bytes for VAX-11 and non-VAX-11 RMS operations.

For all densities, VAX-11 RMS operations require a minimum block size of 14 bytes on output.

The block size on input and output for non-VAX-11 RMS operations can range from 1 to 65,534 bytes when the density is 1600 or 6250 bpi. The VAX/VMS default block size on input and output is 512 bytes for foreign volumes.

You must specify the /BLOCKSIZE qualifier when you are mounting volumes that contain blocks whose size exceeds the default size of 512 bytes. In this case, specify the size of the largest block for the block size.

/DENSITY=n

Specifies the density (in bpi) at which the volume will be written on output. Values for this qualifier are 800, 1600, or 6250 depending on the density that the drive supports. If not specified, density defaults to the density at which the first record on the volume was written. VAX/VMS writes data at the density that you specify with the MOUNT command only if the first operation that you perform on the volume is a write operation.

/FOREIGN
/NOLABEL

Either of these qualifiers indicate that VAX/VMS does not interpret the volume as an ANSI-labeled volume. If you mount a volume with either qualifier, you must supply a program to read and process the volume.

When a volume mounted with either of these qualifiers is being read and the driver encounters a tape mark, VAX/VMS returns an end-of-file condition to your process. By default, VAX/VMS writes data in segments delimited by tape marks. VAX/VMS interprets a tape mark as the end of a data segment indicated by the end-of-file message.

All users have read and write access to a foreign volume. The reason is that the default protection for magnetic tape volumes allows system users and the volume owner read and write access. Because the process from which a foreign volume is mounted owns the volume, whoever mounts a foreign volume is the volume owner.

/OVERRIDE=(option[,...])

Inhibits one or more of the access checks performed by MOUNT. VOLPRO privilege is required to use this qualifier and its options. The volume must be ANSI-labeled.

ACCESSIBILITY — Prevents the access check of the VOL1 and HDR1 accessibility fields. When this option is specified with the EXPIRATION option and you have VOLPRO privilege, this option prevents a runaway tape or timeout condition.

EXPIRATION — Inhibits an access check on the expiration date field in the HDR1 label. Thus, this option overrides volume and file expiration dates. However, when this option is specified with the ACCESSIBILITY option and you have VOLPRO privilege, this option prevents a runaway tape or timeout condition.

/OWNER_UIC=uic

Assigns the user identification code (UIC) that you specify to the volume and its files. Specify the UIC variable in the format:

[g,m]

g is an octal number in the range 0 through 377 and denotes the group number.

m is an octal number in the range 0 through 377 and denotes the member number.

Either square ([]) or angle (<>) brackets are required in the UIC specification.

Use this qualifier to specify a UIC other than the default, the UIC of the current process. You must specify the /PROTECTION qualifier with the /OWNER_UIC qualifier or VAX/VMS writes no UIC to the volume.

/PROTECTION=code

Stores a protection code for the volume in the I/O data base. The protection code is not written to the volume.

You can specify access for group and world user categories. Valid protection codes include read (R), write (W), logical I/O (L), and physical I/O (P); execute and delete access are not applicable to magnetic tape volumes. System users and the volume owner always have RWLP access to magnetic tape volumes, regardless of the protection code that you specify.

/RECORDSIZE=n

Specifies the number of bytes in each record.

Use this qualifier with the /FOREIGN or /NOLABEL and /BLOCKSIZE qualifiers to read or write blocked fixed-length records only. The record size must be less than or equal to the specified or default block size. For example, if you specify /BLOCKSIZE=28 and /RECORDSIZE=7, the maximum size of each block will be 28 bytes and each block will contain no more than four 7-byte records.

5.2.2 Mounting a Volume

The procedure below uses the qualifiers described in Section 5.2.1 to set characteristics for the foreign volume while it is mounted.

1. Specify the ALLOCATE command to allocate the drive before loading the volume. Allocating the drive prior to loading the volume ensures another user cannot preempt the drive. However, you must allocate the drive from the same process from which you will mount the volume. Refer to Section 4.1 for details on the ALLOCATE command.

```
$ ALLOCATE MTA1:
_MTA1:  ALLOCATED
```

VAX/VMS informs you that the drive MTA1 is allocated.

2. Ensure the volume contains a write ring and load the volume on the drive. A volume that does not contain a write ring is write-locked and cannot be written.

3. Press the on-line button if the drive does not go on line automatically. The indicator lamp should light informing you that the drive is on line.

4. Specify the MOUNT command string including qualifiers and a physical or logical device name as shown below (specifying a logical name is optional).

```
$ MOUNT/FOREIGN/DENSITY=1600/BLOCKSIZE=640 -
$_/RECORDSIZE=80 MTA1:  " " GEORGE
%MOUNT-I-MOUNTED, mounted on MTA1:
```

VAX/VMS returns a message informing you that the volume is mounted on the drive MTA1. Although not indicated by the message, VAX/VMS also assigns the logical name GEORGE to the volume mounted on MTA1:

5.2.3 Mounting an Unformatted Volume

To mount an unformatted volume, follow the procedure below. You will need the VOLPRO privilege.

1. Allocate a tape drive before loading the volume.

```
$ ALLOCATE MTA1:
_MTA1:  ALLOCATED
```

2. Ensure that the volume contains a write ring and load the volume on the drive. A volume that does not contain a write ring is write-locked and cannot be written.

3. Load the volume on the drive and place the drive on-line.

4. Specify the MOUNT command string below. Instead of overriding protection, VOLPRO privilege and the command string prevent VAX/VMS from performing access checks on the volume that can cause either a timeout or a runaway tape condition.

```
$ MOUNT/OVERRIDE=(ACCESSIBILITY,EXPIRATION)/NOLABEL MTA1:  " " TESTED
%MOUNT-I-MOUNTED, mounted on _MTA1:
```

VAX/VMS returns a message informing you the volume is mounted on drive MTA1. Although not indicated in the message, VAX/VMS also assigns the logical name TESTED to the volume mounted on the drive. Note that when you mount a volume foreign and specify both the ACCESSIBILITY and EXPIRATION options, VAX/VMS does not return the volume identifier in the message if the volume is ANSI labeled.

5.3 COPYING FILES

You can copy files to or from a foreign volume by using either the COPY command (described in the VAX/VMS Command Language User's Guide) or the Convert Utility (described in the VAX-11 Record Management Services Utilities Reference Manual).

File specifications are not applicable when you copy files to or from foreign volumes. You can copy files to or from a foreign volume if the smallest record within any file is 14 bytes. VAX/VMS writes tape marks to delimit data written on foreign volumes. Specifying the COPY command copies the data between tape marks. However, unlike an ANSI-labeled volume, for a foreign volume you specify only the device name without a file specification, whether in the input or output specification of the COPY command. Each COPY command that you specify copies data between tape marks.

Therefore, you must issue one COPY command for each segment of data that you copy to or from a foreign volume. If any record within a file is smaller than 14 bytes, the copy operation will be incomplete or fail, and your process will receive the following error messages:

```
%COPY-E-WRITEERR, error writing RAFT:[]COP.DAT;1
     -RMS-F-SYS, QIO system service request failed
     -SYSTEM-F-BADPARAM, bad parameter value
```

You can use the Analyze/RMS_File Utility to detect the size of the largest record and the Convert Utility to extend the size of records smaller than 14 bytes and then copy the file to TAPE. All files copied to a foreign volume with the Convert Utility should have the same record size. If you copy files with varying record sizes, you must dismount and remount the volume and respecify the /RECORDSIZE qualifier to read records in a particular data segment. Thus, you must also record the order and record size of each file that you copy to the foreign volume. Accordingly, it is recommended that you select a standard record size such as 80 or 132 bytes. Make sure that the selected record size is as large as the largest record in any file that you copy to a particular volume.

5.3.1 Copying Files from a File Structured Volume

To copy files from a file-structured volume to a non-file-structured foreign volume, follow the procedure below. The procedure copies three files from an ANSI-labeled volume to a foreign volume.

1. Mount the volumes involved in the copy operation.

```
$ MOUNT MTA1:  FRESKI SKI
% MOUNT-I-MOUNTED, FRESKI mounted on _MTA1:

$ MOUNT/FOREIGN MTA0:  " " SKIED
%MOUNT-I-MOUNTED, mounted on _MTA0:
```

2. Specify the COPY command string, including the full device and file name specifications in the input specification, but specify the device name only in the output specification. File names are specified on input because the volume from which files will be copied is ANSI-labeled and file structured. However, only the device name is specified on output because the volume to which the files will be copied is foreign and is not file structured.

```
$ COPY/LOG MTA1:PROG1.EXE, PROG2.EXE, PROG3.EXE MTA0:
%COPY-S-COPIED, MTA1:[]PROG1.EXE;1 copied to MTA0:  (92 records)
%COPY-S-COPIED, MTA1:[]PROG2.EXE;6 copied to MTA0:  (70 records)
%COPY-S-COPIED, MTA1:[]PROG3.EXE;2 copied to MTA0:  (77 records)
%COPY-S-NEWFILES, 3 files created
```

Because no version numbers were specified for the files copied from the ANSI-labeled magnetic tape volume, the first version of each specified file that the MTAACP finds on the the ANSI-labeled volume will be the version of the file that is copied to the foreign volume. When you use the /LOG qualifier with the COPY command, the system will inform you which files were copied and will tally the number of records copied and the number of files created.

5.3.2 Copying Files from a Non-file-structured Volume

The procedure below illustrates how to copy data segments from a non-file-structured foreign volume to a file-structured ANSI-labeled volume. Note that you must specify a COPY command for each segment of data that you copy from a foreign volume to a file-structured volume.

1. Mount the volumes from and to which files will be copied.

```
$ MOUNT/NOLABEL MTA1:  " " COLD:
%MOUNT-I-MOUNTED, mounted on _MTA1:

$ MOUNT MTA0:  FEVER SEASON:
%MOUNT-I-MOUNTED, FEVER mounted on _MTA0:
```

2. Specify a COPY command for each segment on the foreign volume that will be copied to files on the file structured volume. Because the volume to which data segments are copied is file structured, you must specify a file name in the output file specification.

```
$ COPY/LOG MTA1: SEASON:FILE1.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE1.DAT;1 (92 records)
$ COPY/LOG MTA1: SEASON:FILE2.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE2.DAT;1 (62 records)
$ COPY/LOG MTA1: SEASON:FILE3.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE3.DAT;1 (23 records)
$ COPY/LOG MTA1: SEASON:FILE4.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE4.DAT;1 (48 records)
$ COPY/LOG MTA1: SEASON:FILE5.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE5.DAT;1 (37 records)
$ COPY/LOG MTA1: SEASON:FILE6.DAT
%COPY-S-COPIED, MTA1: copied to SEASON:FILE6.DAT;1 (10 records)
```

When the /LOG qualifier is used with the COPY ocmmand, it informs you that the files you specified were copied to the appropriate device, and tallies the number of records transfered.

5.3.3 Copying Files With the Convert Utility to a Foreign Volume

Files containing records smaller than 14 bytes cannot be copied successfully to a foreign volume with the COPY command. However, the Convert Utility (CONVERT) allows you to pad and extend the record size to make it suitable for a foreign volume. In addition, CONVERT itself can copy the file to a foreign volume. The procedure below illustrates how to use the Convert Utility. For more details, refer to the VAX-11 Record Managment Services Utilities Reference Manual.

1. Mount both volumes. Specify the /RECORDSIZE and /BLOCKSIZE qualifiers in addition to either the /NOLABEL or /FOREIGN qualifier when you mount the foreign volume.

```
$ MOUNT/FOREIGN/RECORDSIZE=80/BLOCKSIZE=2400 MTA1:
%MOUNT-I-MOUNTED, mounted on _MTA1:

$ MOUNT MTA0:  LATER TODAY
%MOUNT-I-MOUNTED, LATER mounted on _MTA0:
```

VAX/VMS informs you that the foreign volume is mounted on drive MTA1:, and that the target volume is mounted on MTA0:.

2. Check the size of the largest record within a file. To do so, use the Analyze/RMS_File Utility as shown below.

```
$ ANALYZE/RMS_FILE/STATISTICS TODAY:TICKET.DAT

RMS File Statistics                                14-JUN-1982 21:33:20.07
TODAY:[]TICKET.DAT;3

...

FILE HEADER

        File Spec: TODAY:[]TICKET.DAT;3
        File ID: (1,1,1)
        Owner UIC: [012,141]
        Protection:  System: RWED, Owner: RWED, Group: RWED, World: RWED
        Creation Date:     5-APR-1982 00:00:00.00
        Revision Date:    17-NOV-1858 00:00:00.00, Number: 0
        Expiration Date: none specified
        Backup Date:      none posted
        Contiguity Options:  none
        Performance Options: none
        Reliability Options: none


RMS FILE ATTRIBUTES

        File Organization: sequential
        Record Format: variable
        Record Attributes:   carriage-return
        Maximum Record Size: 0
        Longest Record: 77
        Blocks Allocated: 0, Default Extend Size: 0
        End-of-File VBN: 1, Offset: %X'0000'


The analysis uncovered NO errors.

ANALYZE/RMS_FILE/STATISTICS TODAY:TICKET.DAT


ANALYZE/RMS_FILE/STATISTICS TODAY:TICKET.DAT
```

3. Extend the size of records smaller than 14 bytes to that specified in the MOUNT command and ensure the record size that you specify is sufficient. To do so, specify the following commands:

```
$ CONVERT/PAD/FDL=SYS$INPUT
$_Input:  TODAY:TICKET.DAT
$_Output:  MTA1:
RECORD
SIZE 80
FORMAT FIX I5 <CTRL/Z>
$
```

Specifying the above CONVERT Command string and input and output parameters pads and extends the size of records when copying them from the ANSI-labeled volume mounted on MTA0 to the foreign volume mounted on MTA1. CONVERT command and qualifiers can pad short records to correspond to the record size specified by the MOUNT command in the first step. After you specify the command line, CONVERT prompts you for input and output parameters. To the $_Input: prompt, supply the specification of the file containing the records that were too short. To the $_Output: prompt, specify only the physical or logical device name of the foreign volume. Specifying CTRL/Z both invokes and exits CONVERT. To verify that the file was copied to the foreign volume, you can rewind the foreign volume and type the volume as shown in the steps below.

4. To rewind a foreign volume, specify the SET MAGTAPE/REWIND command string shown below.

```
$ SET MAGTAPE/REWIND MTA1:
$
```

5. To verify the contents of a file, specify the TYPE command as shown below.

```
This is a test file.

Some records contain less than 14 bytes.

This file was copied to a foreign volume with the VAX-11 RMS
Convert Utility.

Any record less than 80 bytes was extended and padded to 80
bytes.
```

The TYPE command types the first data segment on the volume. If you copy multiple files to a foreign volume with the VAX-11 RMS Convert Utility, each file is copied to the volume in a separate data segment. Therefore, you must specify a TYPE command for each data segment copied to the volume until you reach the segment that you want to verify.

5.4 DISMOUNTING VOLUMES

Use the DISMOUNT command described in the VAX/VMS Command Language User's Guide to dismount foreign volumes. A volume will be dismounted and unloaded automatically if either the system fails or you log out of the process that mounted the volume. You can use the procedure in Section 5.2.2 to remount a volume.

The following command dismounts the volume on MTA1:.

```
$ DISMOUNT MTA1:
```

5.5 CREATING, READING, AND WRITING A FOREIGN VOLUME

The command procedures below are written in the DIGITAL Command Language (DCL). Together they allow you to create, read, or write data in a simple user-defined foreign format.

```
$!
$!              FOREIGN.COM
$!
$! This is the master command procedure. It sets up the user account and
$! mounts the volume with the /FOREIGN qualifier. If the user wants to
$! read a foreign volume, the FORREAD.COM command procedure is called.
$! If the user wants to write a foreign volume the FORWRITE.COM
$! command procedure is called.
$!
$       verify_off_on = F$VERIFY ( 0 )
$       ON CONTROL_Y    THEN GOTO clean_up
$       ON WARNING      THEN GOTO clean_up
$
$!
$! If VOLPRO privilege is not set but the user account has SETPRV, VOLPRO is
$! set to allow the user to mount a new unformatted volume.  If VOLPRO
$! privilege cannot be set, the user process is notified that the process
$! has insufficient privilege to write an unformatted volume. The user is also
$! asked whether to continue or exit the procedure.
$!
$       volpro_off_on = F$SETPRV ( "VOLPRO" )
$       IF F$PRIVILEGE ( "VOLPRO" ) THEN GOTO cont
$       WRITE SYS$OUTPUT "Insufficient Privilege to write an unformatted volume!"
$       INQUIRE/NOPUNC continue "Do you wish to continue (Y|N) ? "
$       IF .NOT. continue THEN GOTO clean_up
$ cont:
$
$!
$! Find out where the volume is mounted
$!
$ set_drive:
$       INQUIRE tape_drive "Tape drive"
$       IF tape_drive .EQS. "" THEN GOTO set_drive
$       IF F$LOCATE ( ":", tape_drive ) .EQ. F$LENGTH ( tape_drive ) -
            THEN tape_drive = tape_drive + ":"
$       ASSIGN 'tape_drive' tape
$
$!
$! Try allocating and mounting the volume
$!
$       ALLOCATE tape:
$       MOUNT/NOASSIST/FOREIGN/OVERRIDE=(ACCESSIBILITY,EXPIRATION) tape:
$
$!
$! The user is asked whether a file will be read or written
$!
$
$ read_write:
$       INQUIRE/NOPUNC operation "Read or Write a file ? "
$       IF F$LOCATE ( operation, "READ" )  .EQ. 0 THEN GOTO read_op
$       IF F$LOCATE ( operation, "WRITE" ) .EQ. 0 THEN GOTO write_op
$       GOTO read_write
$
$ read_op:
$       @FORREAD
$       GOTO clean_up
$
$ write_op:
$       @FORWRITE
$
$ clean_up:
$
```

```
$!
$!  Reset the user account the way it was before the command procedure began
$!
$       SET NOON
$       DISMOUNT/NOUNLOAD tape:
$       DEALLOCATE tape:
$       DEASSIGN tape
$       volpro_off_on = F$SETPRV ( volpro_off_on )
$       verify_off_on = F$VERIFY ( verify_off_on )



$!  FORWRITE.COM    Writes Data to a Foreign Volume
$!
$!  This command procedure writes data to a foreign volume.
$!  The data format is as follows:
$!      Each record is a block.
$!      Records are variable length.
$!
$!      The first four characters of a record are digits that are
$!      padded on the left with spaces. This sequence field starts at
$!      1 and increases with by 1 with each record.
$!
$!      The fifth character of a record is a comma.
$!
$!      The sixth through ninth characters are digits that are padded on the
$!      left with spaces. This is a size field, which is the size of the data
$!      field.
$!
$!      The tenth character of a record is a vertical bar.
$!
$!      The rest of the record is data.
$!
$!      Records are padded to be at least 20 characters long.
$!
$
$       seq_num = 0                 ! initialize the sequence number
$       spaces  = "                 "   ! used to pad records less then 20 chars
$
$       OPEN/WRITE tape_file tape: ! open the output file
$ next_line:
$
$       seq_num = seq_num + 1       ! increment the sequence number
```

```
$!  Prompt the user for the data
$!      close the file and exit if ^Z is entered
$!
$       READ/PROMPT="Record # ''seq_num' : "/END_OF_FILE=end_of_input -
            SYS$COMMAND user_data
$
$!  Find the size of the record and the number of pad characters needed
$!  Note a negative number of pad characters does not return any characters
$!
$       data_size = F$LENGTH ( user_data )
$       pad_chars = 10 - data_size
$       IF pad_chars .LT. 0 THEN pad_chars = 0
$
$!  Construct the output record using FAO
$!
$       out_rec = F$FAO ( "!4UL,!4UL!!AS", seq_num, data_size, user_data ) + -
                  F$EXTRACT ( 1, pad_chars, spaces )
$
$       WRITE tape_file out_rec ! write the formatted output record
$       GOTO next_line
$
$ end_of_input:
$       CLOSE tape_file
```

```
$!              FORREAD.COM    Reads a Data from a Foreign Volume
$!
$!  This command procedure is called when data will be read.  The procedure
$!  reads data on a foreign volume.  The data format is defined in
$!  FORWRITE.COM command procedure.
$!
$
$       seq_num = 0                ! initialize the sequence number
$
$       OPEN/READ tape_file tape:  ! open the output file
$ next_line:
$
$       seq_num = seq_num + 1      ! increment the sequence number
$
$       READ/END_OF_FILE=end_of_input tape_file in_rec
$
$       record_num = F$EXTRACT ( 0, 4, in_rec ) - " " - " " - " "
$       record_num = F$INTEGER ( record_num )
$       IF seq_num .NE. record_num THEN -
           WRITE SYS$OUTPUT "Error possible data lost, record sequence broken!"
$
$!
$!  Find the size of the data and extract the data
$!
$       data_size = F$EXTRACT ( 5, 4, in_rec ) - " " - " " - " "
$       data_size = F$INTEGER ( data_size )
$       data_rec  = F$EXTRACT ( 10, data_size, in_rec )
$
$       WRITE SYS$OUTPUT data_rec ! write the data output record
$       GOTO next_line
$
$ end_of_input:
$       CLOSE tape_file
```

APPENDIX A

VAX/VMS ANSI LABELS

Tables in this appendix describe fields in VAX/VMS ANSI volume, header, and trailer labels. Because trailer labels are symmetrical counterparts of header labels and most fields in both label sets contain the same information, separate tables for the trailer labels are not in this appendix. Section 2.7 lists the differences between header and trailer label fields.

Table A-1: VAX/VMS ANSI VOL1 Label

Character Position	Field (length in bytes)	Contents
1 - 3	Label Identifier (3)	Alphabetic characters VOL
4	Label Number (1)	Numeric character 1
5 - 10	Volume Identifier (6)	Volume label consists of ASCII "a" characters.
11	Accessibility (1)	Volume accessibility; provides compatibility with some non-VAX/VMS systems. A space, the VAX/VMS default, indicates no restrictions. To write any ASCII "a" character in this field, use /LABEL=VOLUME_ACCESSIBILITY with the INITIALIZE command. Any character but a space indicates the /OVERRIDE qualifier must be used with the INITIALIZE and MOUNT commands.
12 - 37	Reserved (26)	Spaces
38 - 50	Owner Identifier (13)	Volume ownership set by the INITIALIZE/PROTECTION command. The contents of this field are used for volume protection
51	DIGITAL Standard Version 1	Numeric character 1
52 - 79	Reserved (28)	Spaces
80	Label Standard Version 1	Numeric character 3

Table A-2: First File Header Label (HDR1) Fields

Character Position	Field (length in bytes)	Contents
1 - 3	Label Identifier (3)	Alphabetic characters HDR
4	Label Number (1)	Numeric character 1
5 - 21	File Identifier (17)	A user-supplied file name and file type
22 - 27	File-Set Identifier (6)	Same as the file-set identifier of the first file on the first volume, whether single or multivolume configuration
28 - 31	File-Section Number (4)	Numeric characters starting at 0001 and incrementing by 1 for each additional volume with respect to the first volume on which the file begins
32 - 35	File-Sequence Number (4)	File number within the volume set for this file; consists of numeric characters, starting at 0001 that indicate the position of this file with respect to the first file of the set
36 - 39	Generation Number (4)	Numeric characters that indicate the unique generation of a file
40 - 41	Generation-Version (2) Number	Numeric characters that indicate the version of a particular generation of a file
42 - 47	Creation Date (6)	System stores the date in the Julian format (#YYDDD)[1]; the default is the current date
48 - 53	Expiration Date (6)	User specified Julian date (#YYDDD)* or default is the creation date, indicating file expires immediately
54	Accessibility (1)	File accessibility; provides compatibility with some non-DIGITAL systems. A space (used by DIGITAL systems) indicates no restrictions. Any character but a space indicates the /OVERRIDE qualifier must be used at mount time for the user to access this file.
55 - 60	Block Count (6)	Always 000000 for the HDR1 label

1. The number sign (#) in the Julian format represents a space.

(continued on next page)

Table A-2 (Cont.): First File Header Label (HDR1) Fields

Character Position	Field (length in bytes)	Contents
61 - 73	System Code (13)	Identifies the file system that created the file. DEC, the 3-character constant, occupies positions 61 through 63, followed by the name of the file system; DECFILE11 2 indicates VAX/VMS Version 1.6 and earlier, and DECFILE11A indicates VAX/VMS Version 2.0 and later.
74 - 80	Reserved (7)	Spaces

Table A-3: Second File Header Label (HDR2) Fields

Character Position	Field (length in bytes)	Contents
1 - 3	Label Identifier (3)	Alphabetic characters HDR
4	Label Number (1)	Numeric character 2
5	Record Format (1)	Character definition: F fixed-length D variable-length The S for spanned record format is returned as an undefined format when processed by VAX/VMS[1]
6 - 10	Block Length (5)	Five numeric characters specifying the maximum number of characters per block
11 - 15	Record Length (5)	Numeric characters indicating the record length for fixed-length records or the maximum record length for variable-length records
16	System-Dependent (1)	In VAX/VMS Version 2.1 and later versions, this field contains a space indicating the VAX-11 RMS attributes are in the HDR3 label For VAX/VMS Version 2.0 and previous versions, this field does not contain a space but contains the first byte of the VAX-11 RMS attributes, indicating the VAX-11 RMS attributes are in the HDR2 label

1. To process undefined records properly, the user must know what the original format of the records was. Only logical I/O can be used to process undefined record formats.

(continued on next page)

Table A-3 (Cont.): Second File Header Label (HDR2) Fields

Character Position	Field (length in bytes)	Contents
17 - 36	System-Dependent (20)	Spaces available for future use. For VAX/VMS Version 2.0 and earlier, this field contains the VAX-11 RMS attribute in binary format
37	Form Control (1)	Defines the carriage control applied to the records within a file, as follows: A First byte of record contains FORTRAN control characters M Record contains all form control information Space Line feed/carriage return will be inserted between records (default)
38 - 50	System-Dependent (13)	Spaces available for future use. For VAX/VMS Version 2.0 and earlier, this field contains the VAX-11 RMS attributes in binary format
51 - 52	Buffer Offset (2)	The numeric characters 00
53 - 80	Reserved (28)	Spaces

Table A-4: Third File Header Label (HDR3) Fields

Character Position	Field (length in bytes)	Contents
1 - 3	Label Identifier (3)	Alphabetic characters HDR
4	Label Number (1)	Numeric character 3
5 - 68	VAX-11 RMS Attributes (64)	Files-11 record attributes that override information in fields of the HDR2 label
69 - 80	System-Dependent (12)	Spaces

APPENDIX B

INITIALIZED VAX/VMS ANSI LABELS

This appendix contains the initalization values for VAX/VMS ANSI volume, header, and trailer labels.

Table B-1: Contents of VOL1 Label Upon Initialization

Field	Character Positions	Contents
Label Identifier	1 - 3	VOL
Label Number	4	Numeric character 1
Volume Identifier	5 - 10	Volume identifier
Accessibility	11	An ASCII "a" character, if specified during initialization; otherwise, VAX/VMS defaults a space
Owner Identifier	38 - 50	D&C plus protection mask, if specified when initializing the volume. If not specified, the default is spaces.
DIGITAL Standard Version	51	Numeric character 1
Label Standard Version	80	Numeric character 3

Table B-2: Contents of HDR1 and EOF1 Labels Upon Initialization

Field	Character Positions	Contents
Label Identifier	1 - 3	3 alphabetic characters that identify the label type (HDR or EOF)
Label Number	4	Numeric character 1

(continued on next page)

Table B-2 (Cont.): Contents of HDR1 and EOF1 Labels Upon Initialization

Field	Character Positions	Contents
File Identifier	5 - 21	Spaces
File-Set Identifier	22 - 27	Copy of volume-id
File-Section Number	28 - 31	0000
File-Sequence Number	32 - 35	0000
Generation Number	36 - 39	0001
Generation Version Number	40 - 41	00
Creation Date	42 - 47	Date of initialization
Expiration Date	48 - 53	Date of initialization
Accessibility	54	Space
Block Count	55 - 60	000000
System Code	61 - 73	DECFILE11A
Reserved	74 - 80	Spaces

Table B-3: Contents of HDR2 and EOF2 Labels Upon Initialization

Field	Character Positions	Contents
Label Identifier	1 - 3	3 alphabetic characters that identify the label type (HDR or EOF)
Label Number	4	Numeric character 2
Record Format	5	F
Block Size	6 - 10	00000
Record Length	11 - 15	00000
System-Dependent Fields	16 - 35	Spaces
Form Control	37	Space
System-Dependent Fields	38 - 50	Spaces
Buffer Offset	51 - 52	00
Reserved	53 - 80	Spaces

APPENDIX C

GUIDELINES FOR SELECTING AND HANDLING MAGNETIC TAPE

Magnetic tape is made of one-half-inch mylar tape, coated on the recording side with ferromagnetic iron oxide. Both the mylar and the oxide coating can deteriorate and cause corruption of the stored data when a volume is defective, stored improperly, or exposed to dust, air, magnetic fields, or extreme temperature or humidity.

When you select a type of magnetic tape volume, ensure that both the quality of the mylar and the adhesive used to bond the oxide coating to the mylar is high quality and reliable. If either is defective, either "sticktion" or "shedding" can occur. Sticktion is a condition in which the mylar tape melts due to friction and defective, deteriorated, or poor quality tape. Sticktion is more likely to occur on a high-speed drive during a tape-positioning operation. The friction generated while the drive slows down and stops the tape can melt a defective, deteriorated, or a poor quality tape, causing the mylar to stick to the heads or other components of the drive. Shedding is a condition in which the oxide coating sheds onto the read/write heads, the capstan, or other drive components. If other volumes are mounted on a contaminated drive, the oxide coating can be deposited onto these volumes, corrupting the data that they contain. Data corruption occurs because the contaminant lifts the tape off the head of the drive which causes the signal to be lost from the tape. Shedding is a critical problem with high-speed and high-density drives because the slightest contaminant can cause extensive data corruption or loss.

Improper storage or maintenance of volumes, especially archival volumes that are stored for indefinite or extended periods of time, can accelerate deterioration of the mylar and the adhesive causing sticktion or shedding and loss or corruption of data. Two common problems are brittleness and compression. Brittleness can occur because an archival volume is not recycled to a drive for cleaning and flexing of the tape. Compression usually occurs because the volume is rewound tightly on a high-speed drive before being stored. The pressure exerted on the tightly wound volume compresses the tape toward the hub (center) of the reel.

To avoid data corruption caused by any of the above conditions, you should:

- Always use high quality magnetic tape volumes
- Use a cleaning machine to clean tapes that have been written previously before using them again
- Ensure the heads, the capstan, and other components on the magnetic tape drive are clean and properly adjusted
- Avoid excessive handling of the leader (the tape preceding the BOT marker) with your hands

GUIDELINES FOR SELECTING AND HANDLING MAGNETIC TAPE

- Avoid handling a volume in such a manner as to crimp the tape
- Store volumes at the appropriate temperature and humidity as directed by the manufacturer or specified in the ANSI standard for unrecorded magnetic tape for information interchange
- Periodically mount archival volumes on drives to flex the tape
- Rewind archival volumes at a low speed to prevent compression

APPENDIX D

EXCEPTIONS TO ANSI STANDARD LEVEL 3

The VAX/VMS ANSI-labeled format fully supports Level 2 and is based on Level 3 of the ANSI standard for magnetic tape labels and file structure. This appendix lists the exceptions to the Level 3 ANSI standard.

ANSI optional labels other than the HDR3, EOF3, or EOV3 labels created by VAX/VMS are not supported.

ANSI user labels are not supported. VAX/VMS does not write any user labels (UVLS, UHLS, or UTLS) on output and ignores them on input.

VAX/VMS does not support buffer offset. All data characters are read on input and written on output.

There is a difference in support for file sequence number. When VAX/VMS initializes a magnetic tape volume, a file sequence number of zero is written to the first header label of the first file on the volume to identify this file as the empty dummy file, created during initialization. The ANSI standard specifies that file sequence numbers must have a minimum value of 1. However, when a data file is written to the volume, the empty dummy file is overwritten, and a valid file sequence number is written to this field in the HDR1 label.

GLOSSARY

"a" character

Any character within the subset of ASCII characters called "a" characters. This is the only set of ASCII characters allowed in the labels of ANSI labeled magnetic tape. The set consists of uppercase characters A-Z, numerals 0-9, and the following special characters: space ! " % & ' ( ) * + , - . / : ; < = > ?

ancillary control process (ACP)

An internal software process between a user program and an I/O driver. The MTAACP performs functions supplemental to those done by the magnetic tape driver, such as supporting the processing of ANSI-labeled magnetic tape.

beginning-of-tape marker (BOT)

A reflective piece of tape that physically marks the beginning of the writeable area of a magnetic tape. The ANSI standard requires that a minimum of 14 feet and a maximim of 18 feet of magnetic tape precede the beginning-of-tape (BOT) marker.

block

A group of consecutive bytes of data treated as a unit by the storage medium. A block can contain one or more records.

blocked records

More than one record or parts of records contained in a block.

buffer offset

The length of the auxiliary information that precedes each data block.

end-of-tape marker (EOT)

A reflective piece of tape that physically marks the beginning of the end of the writeable area of a magnetic tape. The ANSI standard requires that a minimum of 25 feet to a maximum of 30 feet of tape, of which 10 feet must be writeable, follow the end-of-tape (EOT) marker.

file

A collection of records pertaining to a single subject. The description, contents, or organization of a file may be arbitrary. A file can be recorded on all or part of a volume, or on more than one volume.

file information block (FIB)

A data structure that is used by $QIO system service routines. The FIB contains much of the information the user must pass to the MTAACP to perform a specific task. The information is supplied by the user. The FIB contains access control values (rewind, current position, read only, write, and so on), file identification, and wildcard context (maintains position for wild card operations). For more information, see the VAX/VMS I/O User's Guide (Volume 2).

file section

A file section is the part of a file that contains the user data and that is delimited by the header and trailer labels. Only one section of a given file can be written on any one volume. Multiple sections of a file or other file sections cannot be interspersed within a file section.

file set

A collection of one or more files recorded consecutively on a volume set.

fixed-length record

A record in a file in which all records are the same size.

foreign tape volume

A volume that is not ANSI-labeled.

label

A record that identifies and delimits a volume or file section. A label identifies the beginning or end of a volume or file.

label group

A collection of one or more contiguous label sets (see also label set).

label identifier

The three initial characters of a label name that identify one or more labels within a label set. These three initial characters will always be the same, such as HDR, which are used to identify header labels within a header label set (see also label set).

label number

A digit that indicates the sequence of a label within a label set. For ANSI labels, label number one is always present if the label set exists (see also label set).

label set

A collection of one or more contiguous labels with the same three initial characters. These three characters are called the "label identifier" (see also label identifier).

record

A set of related data treated as a unit of information. The delineation of a record may be arbitrary and is determined by the designer of the information format.

runaway tape condition

A situation in which a tape spins unceasingly on the drive. A runaway tape condition usually occurs because an operation does not incur a timeout condition. The only way to recover from a runaway tape condition is to take the drive off line.

spanned record

A record that can cross block boundaries. A spanned record consists of one or more data segments. The position of a segment within the record and the length of the segment is denoted by the segment control word (SCW), the first five characters of each segment.

tape mark

A delimiter that consists of the single character block device control character DC3 (which translates to a CTRL/S) that is used to indicate the boundaries between label groups and a file section.

timeout condition

A situation in which an operation must complete in a specified period of time. If the operation does not complete in time, VAX/VMS usually aborts the operation. For magnetic tape volumes, a timeout condition usually occurs because the driver does not find the information being sought.

unblocked records

A record that is contained in a single block and no other records or parts of records are contained in that block.

unformatted volume

A volume which does not contain a logical format. A volume that is purchased from the manufacturer and has never been written or that has been processed by a verifying machine will be unformatted.

variable-length record

A record in a file in which individual records need not be the same size.

volume

A reel of magnetic tape. A volume can contain part of a file, a complete file, more than one file, or sections of more than one file. It does not contain multiple sections of the same file.

volume set

A collection of one or more related volumes on which one and only one file set is recorded.

INDEX

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify) _____

Name _____ Date _____

Organization _____

Street _____

City _____ State _____ Zip Code or Country _____

— Do Not Tear - Fold Here and Tape —

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

BSSG PUBLICATIONS ZK1-3/J35
DIGITAL EQUIPMENT CORPORATION
110 SPIT BROOK ROAD
NASHUA, NEW HAMPSHIRE 03061

— Do Not Tear - Fold Here —

Cut Along Dotted Line